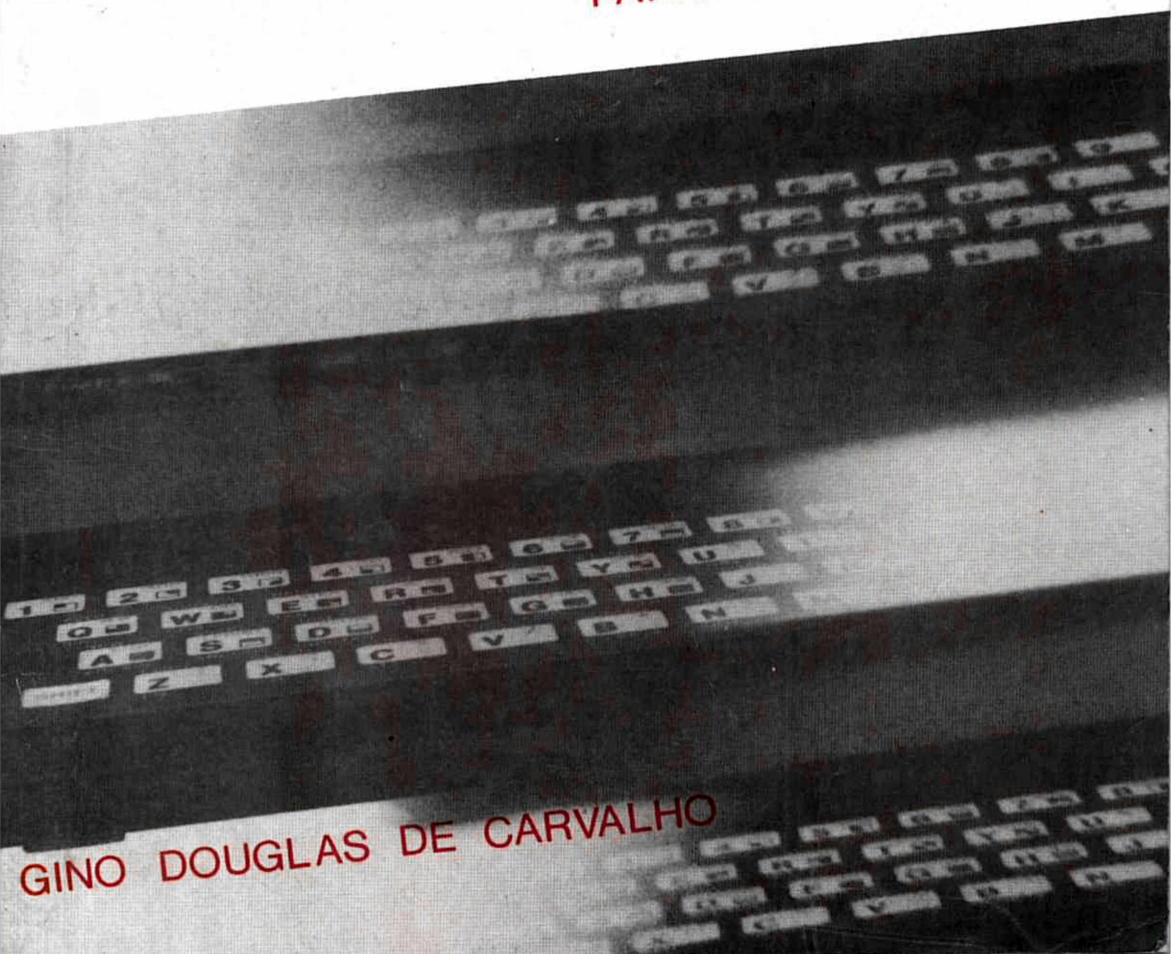
40 ROTINAS EM LINGUAGEM DE MÁQUINA

PARA TK E CP 200

GINO DOUGLAS DE CARVALHO

40 ROTINAS EM LINGUAGEM DE MAQUINA

por Gino Douglas de Carvalho

Primeira Edicao- Marco 1985

Capa por Roberto Amaral Cunha

Impresso por J.A.C. Grafica LTDA.

Editado e distribuido por
Micron Eletronica Com. Ind. Ltda.
Sao Jose dos Campos, SP, Brasil

Composicao pelo Autor

INDICE

P R E F A C I O

Este livro foi escrito pensando-se em duas categorias de leitores, a primeira formada por aqueles que estao se iniciando neste fascinante mundo dos computadores com logica SINCLAIR, e a segunda formada por pessoas que ja dominam totalmente o basic e ja se aventuram em linguagem de maquina. Se voce estiver na primeira categoria, achara aqui varias rotinas prontas para serem utilizadas, bastando ler com atencao as instrucoes contidas na descricao, condicoes de entrada e condicoes de saida, e colocar a rotina no micro atravez da listagem hexadecimal que e encontrada ao final de cada rotina. Para os fanaticos que se enquadram na segunda categoria, a listagem assembly servira de fonte para estudos, possibilitando a qualquer um aprofundar-se no mundo da linguagem de maquina. Como programacao e uma atividade dinamica em constante evolucao, sinta-se livre para efetuar as modificacoes que achar necessaria para melhor enquadrar as rotinas aqui apresentadas aos seus interesses.

Todos os cuidados foram tomados para que as rotinas aqui apresentadas estejam livres de erro, tanto de logica como de impressao, mas como nao somos infaliveis, qualquer comunicacao a respeito de falhas sera muito bem vinda, assim como contatos sobre as duvidas que poderao existir sobre a utilizacao das rotinas.

Existem inumeras pessoas que colaboraram para a execucao deste livro, porem gostaria de ressaltar a participao do Eng. Jose Maria Villera Chagas, que prestou inestimavel ajuda na revisao dos originais, do Eng. Luiz Antonio Nunes, pela cessao dos equipamentos em que foram processados os originais, e do Sr. Delio Santos Lima, pela paciencia demonstrada ate que este livro pudesse lhe ser entregue para edicao.

Os puristas da lingua portugesa irao estranhar a completa ausencia de sinais de acentuacao durante todo o livro, porem pedimos a compreensao de todos, pois o equipamento utilizado no processamento de texto, apesar de construido no Brasil, e de projeto americano e nao tem capacidade para colocao desses sinais. Para as pessoas ja acostumadas com computadores esta ausencia nao causara muitos transtornos, porem esperamos em futuro proximo fazer a redacao como manda o figurino.

Gino Douglas de Carvalho
Caixa Postal 100, CEP 12.200, Sao Jose dos Campos.
Sao Paulo - Brasil.

CAPITULO 1

FUNDAMENTOS DE PROGRAMACAO EM LINGUAGEM DE MAQUINA

INTRODUCAO

O MICROPROCESSADOR Z-80

OS REGISTROS DO Z-80

AS INSTRUCOES DO Z-80

MODOS DE ENDERECAMENTO

INTRODUCAO

Neste capitulo vamos falar sobre alguns fundamentos de programacao em linguagem de maquina. Nao e necessario que voce entenda tudo neste capitulo, ou mesmo que voce o leia, para que possa se utilizar das rotinas contidas neste livro. Porem se voce o fizer tera uma clara visao do que e programacao em linguagem de maquina do Z-80.

O MICROPROCESSADOR Z-80

O microprocessador utilizado nos microcomputadores da linha SINCLAIR e o Z-80, que e um microprocessador de terceira geracao e verdadeiramente um computador em um chip. Diferente do basic, o Z-80 processa as instrucoes no nivel mais rudimentar. Instrucoes tipicas sao somar ou subtrair dois numeros de 8 bits, transferir o conteudo de um registro da CPU para uma localizacao de memoria ou vice-versa. Todo programa e formado por instrucoes do Z-80 colocadas sequencialmente na memoria. Estas instrucoes podem ser de 1 a 4 bytes de comprimento.

OS REGISTROS DO Z-80

Antes de vermos as instrucoes do Z-80, vamos estudar a sua arquitetura. A fig. 1 mostra os registros internos do Z-80 que estao disponiveis para o programador. Os registros sao localizacoes de memoria de acesso rapido e localizados dentro do Z-80. Estes registros sao utilizados para armazenar resultados temporarios e para o processamento

O registro A e o acumulador. Ele contem um dos operandos para as somas, subtracoes e outras operacoes de aritmeticas de 8 bits, enquanto o outro operando pode estar em outro registro ou em uma localizacao de memoria. O resultado destas operacoes tambem estara no acumulador, apos a operacao.

Os registros B, C, D, E, H, e L sao registros de uso geral de 8 bits e podem ser agrupados, formando os registros BC, DE e HL de 16 bits. O par HL e o acumulador para as operacoes de 16 bits, similar ao acumulador A.

Os registros IX e IY sao registros de 16 bits e sao utilizados como registros de indice, ou seja funcionam como ponteiros para localizacoes de memoria.

O registro PC, contador de programa, e o principal registro de controle do Z-80. Ele controla a execucao de todos os programas, seja ele em basic ou em linguagem de maquina. O PC e um registro de 16 bits e aponta sempre para o primeiro byte da proxima instrucao a ser executada.

Conforme o programa vai sendo executado o PC e incrementado para apontar sempre para a proxima instrucao. O PC pode ser modificado por algumas instrucoes, como por exemplo a instrucao JUMP (equivalente a instrucao GOTO em basic).

O SP, ponteiro do stack, e um registro de 16 bits que aponta para a area do stack. A area do stack, ou stack simplesmente, e uma parte da memoria RAM que e reservada para armazenar enderecos de retorno para as instrucoes CALL e tambem resultados temporarios. Este stack cresce para baixo conforme e utilizado, ou seja, o inicio do stack esta em localizacoes de memoria mais elevadas que o seu final. Toda vez que um programa em linguagem de maquina executa uma instrucao CALL (similar ao GOSUB do basic) o endereco de retorno e retirado do PC e colocado no stack. Quando se executa uma instrucao RET (RETURN do basic) o endereco e retirado do stack e colocado no PC.

O registro R foi projetado para fazer o "refresh" de memorias dinamicas, sendo um registro de 7 bits. O registro I e utilizado para alguns modos especiais de interrupcao. Nos micros com logica SINCLAIR estes registros sao utilizados para finalidades diversas da qual foram projetados, e praticamente nao sao utilizados pelos programas em linguagem de maquina.

O registro F e uma reuniao de 8 flags, ou bandeiras, conforme mostra a figura 2.

```
---------------------------------------
* S * Z * X * H * X *P/V* N * C *
---------------------------------------

        S   :Sign flag
        Z   :Zero flag
        H   :Half carry flag
        P/V:Parity(P)/Overflow(V) flag
        N   :Add/Subtract flag
        C   :Carry flag
        X   :Nao utilizado
```

FIG 2-O REGISTRO F (FLAGS)

Flags sao condicoes que sao modificadas pelas instrucoes, podendo ser testadas pelo programa para tomadas de decisoes.

O flag S, sign flag (flag de sinal), e setado (colocado com o valor 1) quando o resultado da operacao for negativo, e resetado (colocado com o valor 0) se o resultado for zero ou positivo.

O flag Z, zero flag (flag de zero), e setado quando o resultado da operacao for zero, e resetado caso contrario.

O flag H, half carry flag (flag do meio carry), armazena o carry (vai um) do bit 3 para o bit 4 do

acumulador, e utilizado somente em algumas instrucoes que trabalham com numeros BCD, binario codificado em decimal.

O flag P/V, parity/overflow flag (flag de paridade e sobrecarga), pode indicar, conforme o tipo de instrucao, se a paridade do resultado e par ou impar, ou se houve sobrecarga, ou seja, o resultado e maior que a capacidade do registro.

O flag N, add/subtract carry (flag de soma ou subtracao), indica se a ultima operacao foi uma adicao ou uma subtracao, e tambem e utilizado somente por instrucoes que trabalham com numeros BCD.

O flag C, carry flag (flag do carry ou vai um), e setado quando houver o vai um no resultado da operacao, e resetado caso contrario.

O registro F e considerado como par do registro A para efeito de armazenamento no stack e para outras operacoes.

Os sete registros de 8 bits A, B, C, D, E, H, L e o registro F sao duplicados no Z-80. Existe um outro conjunto de registros chamados de A', B', C', D', E', H', L' e F', mas somente um dos conjuntos, o principal ou o alternativo, pode estar ativo ao mesmo tempo, podendo o conjunto que nao estiver ativo ser utilizado como armazenamento. Existem duas instrucoes para se acessar o conjunto alternativo, que sao EX AF,AF' para inverter o par AF, e EXX para inverter os pares BC, DE e HL.

Nem todos estes registros que acabamos de ver podem ser utilizados pelo programador, pois alguns deles tem uma funçao especifica nos microcomputadores com logica SINCLAIR. Os registros R e I sao utilizados para gerar a imagem no video, o registro R sendo utilizado como contador de tempo dos pulsos de sincronismo horizontal, e o registro I para acessar a tabela de caracteres arquivada na memoria ROM. Quando se utiliza o modo SLOW os registros IX e AF' sao utilizados por este circuito e nao podem ser alterados sem que corramos o risco de perder o controle do microcomputador . O registro IY e mantido com o valor 4000h para que os programas da ROM possam acessar as variaveis do sistema.

AS INSTRUCOES DO Z-80

O conjunto de instrucoes do Z-80 contem mais de 700 instrucoes diferentes, porem felizmente elas podem ser consideradas como pertencentes a 11 grupos, e portanto e muito mais facil estudarmos estes grupos do que as instrucoes isoladas.

As seguintes abreviacoes serao utilizadas nas descricoes das instrucoes:

r registro simples: A, B, C, D, E, H ou L.
IR registro de indice: IX ou IY
(IR+d) o conteudo do endereco determinado pela soma de d com o registro de indice.

CONJUNTO PRINCIPAL	
A ACUMULADOR	F FLAGS
B	C
D	E
H	L

CONJUNTO ALTERNATIVO	
A' ACUMULADOR	F' FLAGS
B'	C'
D'	E'
H'	L'

IX REGISTRO DE INDICE
IY REGISTRO DE INDICE
S P PONTEIRO DO STACK
P C CONTADOR DO PROGRAMA

I	R

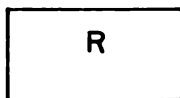
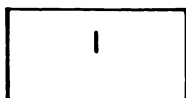



FIG. 1 - OS REGISTROS DO Z-80

s	um operando que pode ser um dos seguintes: r, n, (HL) ou (IR+d).
dd	par de registro: BC, DE, HL ou SP.
qq	par de registro: BC, DE, HL ou AF.
pp	par de registro: BC, DE, HL, IX ou IY.
n	um byte contido na propria instrucao.
(n)	nas instrucoes de input e output, um byte contido na propria instrucao, cujo valor seleciona a porta de I/O.
nn	dois bytes contidos na propria instrucao.
(nn)	dois bytes contidos na propria instrucao, referente a um endereco da memoria.
e	nas instrucoes de JUMP relativo, o valor a ser somado ao valor corrente do PC para determinar o endereco do desvio.
p	nas instrucoes RST (restart), endereco da localizacao de memoria chamado, entre 00h e 38h em multiplos de 8.
b	bit: 0, 1, 2, 3, 4, 5, 6 ou 7.
cc	codigo de condicao,para testar os flags: NZ, Z, NC, C, PO, PE, P, M.
c	codigo de condicao para JUMP relativo: NZ, Z, NC, C.
(HL)	o conteudo da posicao de memoria apontado pelo par HL. Similarmente para (BC) e (DE).

Para os flags sao utilizadas as seguintes abreviacoes:

- * flag nao afetado
- 0 flag resetado (colocado com o valor 0)
- 1 flag setado (colocado com o valor 1)
- x desconhecido como estara o flag
- R flag ajustado de acordo com o resultado
- V para o flag P/V mostra que o flag indica sobrecarga (overflow)
- P para o flag P/V mostra que o flag indica paridade

GRUPO DE ARITMETICA E LOGICA DE 8 BITS: Estas instrucoes executam operacoes aritmeticas e logicas em operandos de 8 bits. Com excecao das instrucoes INC (incremento) e DEC (decremento) todas as instrucoes sao executadas com o acumulador, mesmo que este nao esteja indicado na instrucao. Os flags sao modificados conforme cada uma das instrucoes e o carry flag e utilizado em algumas operacoes.

INSTRUCAO	BYTES	FUNCAO	C	Z	P/V	S	N	H
ADD A,r	1	A<=A+r	R	R	V	R	0	R
ADD A,n	2	A<=A+n	R	R	V	R	0	R
ADD A,(HL)	1	A<=A+(HL)	R	R	V	R	0	R

INSTRUCAO	BYTES	FUNCAO	C	Z	P/V	S	N	H
ADD A,(IR+d)	3	A<=A+(IR+d)	R	R	V	R	0	R
ADC A,s	1-3	A<=A+s+CY	R	R	V	R	0	R
SUB s	1-3	A<=A-s	R	R	V	R	1	R
SBC s	1-3	A<=A-s-CY	R	R	V	R	1	R
AND s	1-3	A<=A AND s	0	R	P	R	0	1
OR s	1-3	A<=A OR s	0	R	P	R	0	0
XOR s	1-3	A<=A XOR s	0	R	P	R	0	0
CP s	1-3	A-s	R	R	V	R	1	R
INC r	1	r<=r+1	*	R	V	R	0	R
INC (HL)	1	(HL)<=(HL)+1	*	R	V	R	0	R
INC (IR+d)	3	(IR+d)<= (IR+d)+1	*	R	V	R	0	R
DEC r	1	r<=r-1	*	R	V	R	1	R
DEC (HL)	1	(HL)<=(HL)-1	*	R	V	R	1	R
DEC (IR+d)	3	(IR+d)<= (IR+d)-1	*	R	V	R	1	R

GRUPO DE ARITMETICA DE 16 BITS: Estas instrucoes executam operacoes aritmeticas com 16 bits. Para a maioria destas operacoes o registro HL e utilizado como acumulador, similar ao uso do registro A em operacoes de 8 bits. Isto significa que HL e utilizado para conter um dos operandos e tambem o resultado da operacao. Os registros de indice IX e IY tambem podem ser utilizados desta maneira para adicoes.

INSTRUCAO	BYTES	FUNCAO	C	Z	P/V	S	N	H
ADD HL,ss	1	HL<=HL+ss	R	*	*	*	0	X
ADC HL,ss	2	HL<=HL+ss+CY	R	R	V	R	0	X
SBC HL,ss	2	HL<=HL-ss-CY	R	R	V	R	1	X
ADD IR,pp	2	IR<=IR+pp	R	*	*	*	0	X
INC ss	1	ss<=ss+1	*	*	*	*	*	*
INC IR	2	IR<=IR+1	*	*	*	*	*	*
DEC ss	1	ss<=ss-1	*	*	*	*	*	*
DEC IR	2	IR<=IR-1	*	*	*	*	*	*

GRUPO DE ARITMETICA DE USO GERAL E CONTROLE DA CPU: Este grupo inclui uma miscelanea de instrucoes, algumas aritmeticas e outras de controle da CPU.

INST.	BYT.	FUNCAO	C	Z	P/V	S	N	H
DAA	1	ajuste decimal acumul.	R	R	P	R	*	R
CPL	1	complementa acumulador	*	*	*	*	1	1
NEG	2	comp. de dois acumul.	R	R	V	R	1	R
CCF	1	comp. carry flag	R	*	*	*	0	X
SCF	1	seta carry flag CY<=1	1	*	*	*	0	0
NOP	1	nao operacao	*	*	*	*	*	*

INST.	BYT.	FUNCAO	C	Z	P/V	S	N	H
HALT	1	suspende operacao CPU	*	*	*	*	*	*
DI	1	desabilita interrupcoes	*	*	*	*	*	*
EI	1	abilita interrupcoes	*	*	*	*	*	*
IM 0	2	modo de interrupcao 0	*	*	*	*	*	*
IM 1	2	modo de interrupcao 1	*	*	*	*	*	*
IM 2	2	modo de interrupcao 2	*	*	*	*	*	*

GRUPO DE LOAD DE 8 BITS: Todas as instrucoes deste grupo transferem um byte entre dois registros da CPU, ou entre um registro e uma localizacao de memoria. Estas instrucoes nao afetam os flags com excecao das instrucoes LD A,I e LD A,R.

INSTRUCAO	BYTES	FUNCAO	C	Z	P/V	S	N	H
LD r,r'	1	r<=r'	*	*	*	*	*	*
LD r,n	2	r<=n	*	*	*	*	*	*
LD r,(HL)	1	r<=(HL)	*	*	*	*	*	*
LD r,(IR+d)	3	r<=(IR+d)	*	*	*	*	*	*
LD (HL),r	1	(HL)<=r	*	*	*	*	*	*
LD (IR+d),r	3	(IR+d)<=r	*	*	*	*	*	*
LD (HL),n	2	(HL)<=n	*	*	*	*	*	*
LD A,(BC)	1	A<=(BC)	*	*	*	*	*	*
LD A,(DE)	1	A<=(DE)	*	*	*	*	*	*
LD A,(nn)	3	A<=(nn)	*	*	*	*	*	*
LD (BC),A	1	(BC)<=A	*	*	*	*	*	*
LD (DE),A	1	(DE)<=A	*	*	*	*	*	*
LD (nn),A	3	(nn)<=A	*	*	*	*	*	*
LD A,I	2	A<=I	*	R	IFF	R	0	0
LD A,R	2	A<=R	*	R	IFF	R	0	0
LD I,A	2	I<=A	*	*	*	*	*	*
LD R,A	2	R<=A	*	*	*	*	*	*

Nota: IFF significa interrupt flip-flop.

GRUPO DE LOAD DE 16 BITS: Estas instrucoes sao similares ao do grupo de load de 8 bits, com a diferenca que agora 16 bits sao envolvidos na transferencia. Nenhum flag e afetado por estas instrucoes e estes flags nao sao mostrados na tabela. E importante notar que nao existe instrucoes para transferencia direta entre pares de registros da CPU.

INSTRUCOES	BYTES	FUNCAO
LD dd,nn	3	dd<=nn
LD IR,nn	4	IR<=nn
LD HL,(nn)	3	HL<=(nn)
LD dd,(nn)	4	dd<=(nn)
LD IR,(nn)	4	IR<=(nn)

INSTRUCOES	BYTES	FUNCAO
LD (nn),HL	3	(nn)<=HL
LD (nn),dd	4	(nn)<=dd
LD (nn),IR	4	(nn)<=IR
LD SP,HL	1	SP<=HL
LD SP,HL	2	SP<=IR
PUSH qq	1	(SP-2)<=qq(l)
		(SP-1)<=qq(h)
		SP<=SP-2
PUSH IR	2	(SP-2)<=IR(l)
		(SP-1)<=IR(h)
		SP<=SP-2
POP qq	1	qq(h)<=(SP+1)
		qq(l)<=(SP)
		SP<=SP+2
POP IR	2	IR(h)<=(SP+1)
		IR(l)<=(SP)
		SP<=SP+2

GRUPO DE TROCA, TRANSFERENCIA E BUSCA DE BLOCOS: Este grupo envolve dois tipos diferentes de instrucoes, as instrucoes de troca (exchange) que comutam dois operandos, e as instrucoes de transferencia e busca de blocos que movem e comparam grandes blocos de dados.

INSTRUCOES	BYTES	FUNCAO	C	Z	P/V	S	N	H
EX DE,HL	1	DE<=>HL	*	*	*	*	*	*
EX AF,AF'	1	AF<=>AF'	*	*	*	*	*	*
EXX	1	BC<=>BC'						
		DE<=>DE'						
		HL<=>HL'	*	*	*	*	*	*
EX (SP),HL	1	(SP)<=>HL	*	*	*	*	*	*
EX (SP),IR	2	(SP)<=>IR	*	*	*	*	*	*
LDI	2	(DE)<=(HL)						
		DE<=DE+1						
		HL<=HL+1						
		BC<=BC-1	*	*	(1)	*	0	0
LDIR	2	(DE)<=(HL)						
		DE<=DE+1						
		HL<=HL+1						
		BC<=BC-1						
		repete ate						
		BC=0	*	*	0	*	0	0
LDD	2	(DE)<=(HL)						
		DE<=DE-1						
		HL<=HL-1						
		BC<=BC-1	*	*	(1)	*	0	0

INSTRUCOES	BYTES	FUNCAO	C	Z	P/V	S	N	H
LDDR	2	(DE)<=(HL)						
		DE<=DE-1						
		HL<=HL-1						
		BC<=BC-1						
		repete ate						
		BC=0	*	*	0	*	0	0
CPI	2	A-(HL)						
		HL<=HL+1						
		BC<=BC-1	*	(2)	(1)	R	1	R
CPIR	2	A-(HL)						
		HL<=HL+1						
		BC<=BC-1						
		repete ate						
		A=(HL) ou						
		BC=0	*	(2)	(1)	R	1	R
CPD	2	A-(HL)						
		HL<=HL-1						
		BC<=BC-1	*	(2)	(1)	R	1	R
CPDR	2	A-(HL)						
		HL<=HL-1						
		BC<=BC-1						
		repete ate						
		A=(HL) ou						
		BC=0	*	(2)	(1)	R	1	R

Notas: (1)- P/V e 0 se BC-1=0, ou 1 caso contrario
(2)- Z e 1 se A=(HL), ou 0 caso contrario

GRUPO DE SET, RESET E TESTE DE BITS: Estas instrucoes trabalham com bits em qualquer um dos registros. A instrucao SET coloca o bit especificado no valor 1, a instrucao RESET coloca 0 no bit especificado, e a instrucao BIT testa se o bit especificado e 0 ou 1.

INSTRUCOES	BYTES	C	Z	P/V	S	N	H
BIT b,r	2	*	R	X	X	0	1
BIT b,(HL)	2	*	R	X	X	0	1
BIT b,(IR+d)	4	*	R	X	X	0	1
SET b,r	2	*	*	*	*	*	*
SET b,(HL)	2	*	*	*	*	*	*
SET b,(IR+d)	4	*	*	*	*	*	*
RES b,s	2-4	*	*	*	*	*	*

GRUPO DE JUMP: Estas instrucoes desviam o programa para a localizacao indicada,frequentemente dependendo das condicoes dos flags. Algumas vezes o endereco esta contido na propria instrucao. No caso de JUMP relativo o endereco de desvio e calculado somando o deslocamento especificado na

instrucao com o conteudo do PC. Nenhuma destas instrucoes afeta os flags.

INSTRUCAO	BYTES	FUNCAO
JP nn	3	PC<=nn
JP cc,nn	3	se cc e verdadeiro PC<=nn, senao continua
JR e	2	PC<=PC+e
JR c,e	2	se c e verdadeiro PC<=PC+e senao continua
JP (HL)	1	PC<=(HL)
JP (IR)	2	PC<=(IR)
DJNZ e	2	B<=B-1 se B=0 continua, senao PC<=PC+e

GRUPO DE CHAMADA E RETORNO DE SUBROTINA: As instrucoes de chamada de subrotina CALL colocam o conteudo do PC no stack, e colocam no PC o endereco contido na instrucao. As instrucoes de retorno RET retiram o conteudo do topo do stack e colocam no PC,reassumindo a execucao do programa apos a instrucao CALL. As instrucoes restart RST sao identicas ao CALL,exceto que a localizacao e especificada em 3 bits da propria instrucao, permitindo somente 8 enderecos entre 00h e 38h. Nenhuma destas instrucoes afetam os flags.

INSTRUCAO	BYTES	FUNCAO
CALL nn	3	(SP-1)<=PC(h) (SP-2)<=PC(1) PC<=nn
CALL cc,nn	3	se cc e verdadeiro PC<=nn, senao continua
RET	1	PC(1)<=(SP) PC(h)<=(SP+1)
RET cc	1	se cc e falso continua se cc e verdadeiro, mesmo que RET
RETI	2	retorno de interrupcao
RETN	2	retorno de interrupcao nao mascarada
RST p	1	(SP-1)<=PC(h) (SP-2)<=PC(1) PC(h)<=0 PC(1)<=p

GRUPO DE INPUT E OUTPUT: Estas instrucoes transferem dados entre um registro da CPU e dispositivos de entrada e saida. Algumas instrucoes transferem apenas um byte de cada vez, enquanto outras transferem blocos inteiros.

INSTRUCAO	BYTES	FUNCAO	C	Z	P/V	S	N	H
IN A,(n)	2	A<=(n)	*	*	*	*	*	*
IN r,(C)	2	r<=(C)	*	R	P	R	0	R

INSTRUCAO	BYTES	FUNCAO	C	Z	P/V	S	N	H
INI	2	(HL)<=(C)						
		B<=B-1						
		HL<=HL+1	X	(1)	X	X	1	X
INIR	2	(HL)<=(C)						
		B<=B-1						
		HL<=HL+1						
		repete ate						
		B=0	X	1	X	X	1	X
IND	2	(HL)<=(C)						
		B<=B-1						
		HL<=HL-1	X	(1)	X	X	1	X
INDR	2	(HL)<=(C)						
		B<=B-1						
		HL<=HL-1						
		repete ate						
		B=0	X	1	X	X	1	X
OUT (n),A	2	A<=(n)	*	*	*	*	*	*
OUT (C),r	2	(C)<=r	*	*	*	*	*	*
OUTI	2	(C)<=(HL)						
		B<=B-1						
		HL<=HL+1	X	(1)	X	X	1	X
OUTIR	2	(C)<=(HL)						
		B<=B-1						
		HL<=HL+1						
		repete ate						
		B=0	X	1	X	X	1	X
OUTD	2	(C)<=(HL)						
		B<=B-1						
		HL<=HL-1	X	(1)	X	X	1	X
OUTDR	2	(C)<=(HL)						
		B<=B-1						
		HL<=HL-1						
		repete ate						
		B=0	X	1	X	X	1	X

Nota: (1)- Se o resultado de B-1=0 o flag Z e setado, em caso contrario resetado

GRUPO DE ROTACAO E DESLOCAMENTO: Estas instrucoes incluem um grande numero de operacoes que rodam e deslocam registros. Ha varias redundancias entre elas porque existem instrucoes compativeis com o microprocessador 8080, que executam operacoes somente no acumulador, e instrucoes exclusivas do Z-80, que operam em todos os registros. Todas as rotacoes e deslocamentos movem o registro indicado somente 1 bit. O deslocamento move cada bit no registro para o proximo bit, a esquerda ou a direita, e preenche o bit vazio com zero. A rotacao move o bit que seria jogado para fora no bit vazio. Todas estas instrucoes sao complicadas pelo modo em que o carry flag participa da operacao. Ha instrucoes de 8 bits em que o bit e movido tanto para o registro como para o carry

flag, e existem instrucoes de 9 bits nas quais o carry flag e utilizado como um bit extra. Os flags N e H sao resetados por estas instrucoes e o flag P/V indica paridade para as instrucoes exclusivas do Z-80. Para uma melhor vizualizacao das funcoes, elas estao mostradas graficamente na figura 3.

INSTRUCAO	BYTES	C	Z	P/V	S	N	H
RLCA	1	R	*	*	*	O	O
RLA	1	R	*	*	*	O	O
RRCA	1	R	*	*	*	O	O
RRA	1	R	*	*	*	O	O
RLC r	2	R	R	P	R	O	O
RLC (HL)	2	R	R	P	R	O	O
RLC (IR+d)	4	R	R	P	R	O	O
RL s	2-4	R	R	P	R	O	O
RRC s	2-4	R	R	P	R	O	O
RR s	2-4	R	R	P	R	O	O
SLA s	2-4	R	R	P	R	O	O
SRA s	2-4	R	R	P	R	O	O
SRL s	2-4	R	R	P	R	O	O
RLD	2	*	R	P	R	O	O
RRD	2	*	R	P	R	O	O

MODOS DE ENDERECAMENTO

Modos de enderecamento resumem todas as maneiras nas quais as instrucoes podem ser executadas no computador. Para executar qualquer operacao o computador deve conhecer o endereco da localizacao envolvida. Existem dez modos de enderecamento descritos abaixo:

1- IMEDIATO: Um byte contido na propria instrucao e movido para um registro. A instrucao tem um comprimento de 2 bytes. Um exemplo seria LD A,1 que carrega o acumulador A com o valor 1.

2- IMEDIATO ESTENDIDO: Igual ao anterior, exceto que dois bytes sao movidos para um par de registros. Comprimento da instrucao de 3 bytes. Um exemplo seria LD HL,1234, que carrega o par HL com o valor 1234.

3- RELATIVO: Aplicado somente para as instrucoes de jump relativo JR. O valor contido na propria instrucao e somado ao PC para determinar o endereco efetivo. O valor deve estar na faixa de -128 a 127. O comprimento da instrucao e 2 bytes e um exemplo seria JR 02, que vai determinar um endereco 4 bytes apos o inicio da instrucao JR (O valor fornecido foi 2 , porem a instrucao JR tem dois bytes e o PC sempre aponta para o incio da proxima instrucao).

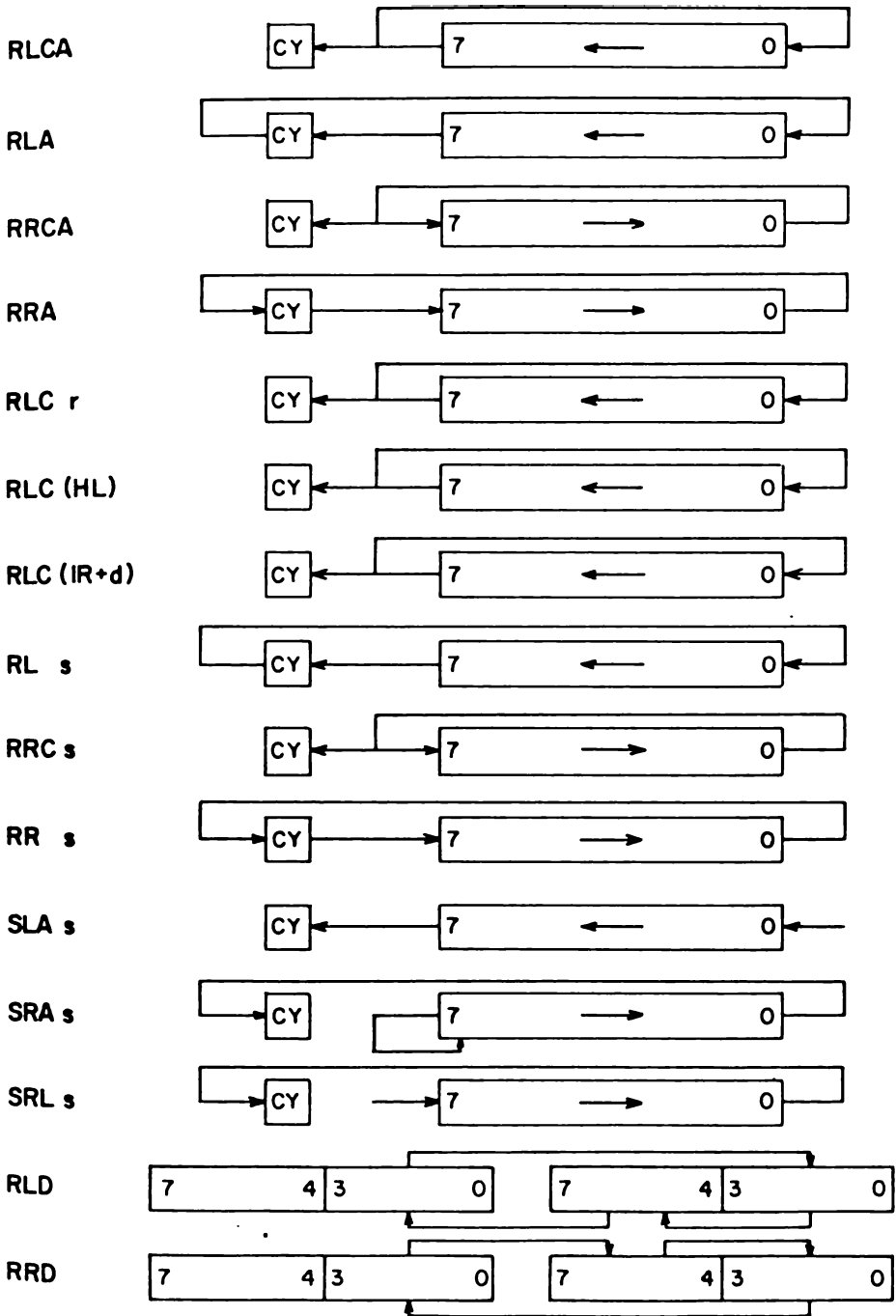



FIG. 3-ROTAÇÕES E DESLOCAMENTOS

4- ESTENDIDO: O endereco do operando e especificado na instrucao. Comprimento de 3 bytes. Um exemplo seria LD A,(4000) que carrega o acumulador com o valor que esta na localizacao de memoria 4000h.

5- INDEXADO: O endereco do operando e determinado pela soma de um byte contido na instrucao com o valor contido em um dos registros de indice. Comprimento de 3 ou 4 bytes. Um exemplo seria LD A,(IY+5), que caso IY esteja com o valor 4000h, carregara o acumulador com o conteudo da localizacao de memoria 4005h.

6- REGISTRO: Um registro e carregado a partir de outro. Comprimento 1 byte. Um exemplo seria LD B,C que carrega o registro B com o valor que esta no registro C.

7- IMPLICADO: Similar ao anterior somente que um dos operando esta subentendido. Comprimento de 1 ou 2 bytes. Um exemplo seria AND C que fara a operacao AND entre o acumulador (nao citado na instrucao) e o registro C.

8- REGISTRO INDIRETO: O endereco de um operando estara contido em um par de registros (BC, DE ou HL). Comprimento de 1 byte. Um exemplo seria LD A,(BC) que carregara o acumulador com o valor da localizacao de memoria cujo endereco esta em BC.

9- BIT: Indica um bit individual em um registro. Comprimento 2 bytes. Um exemplo seria RES 7,C que resetara o bit 7 do registro C.

10- PAGINA ZERO MODIFICADA: Aplica-se somente para as instrucoes restart RST. Somente 3 bits da instrucao sao utilizados para indicar o endereco entre 00h e 38h em multiplos de 8. Comprimento de 1 byte. Um exemplo seria RST 08 que seria um CALL para a localizacao 0008h.

CAPITULO 2

UTILIZANDO LINGUAGEM DE MAQUINA

INTRODUCAO

LINGUGEM DE MAQUINA E LINGUAGEM ASSEMBLY

ONDE COLOCAR AS ROTINAS

COMO PASSAR DADOS PARAS AS ROTINAS

UM PROGRAMA MONITOR

INTRODUCAO

Neste capitulo vamos descrever como entender e utilizar as rotinas deste livro. Portanto e fundamental que antes de se utilizar qualquer rotina este capitulo seja lido e compreendido. Comecaremos comentando sobre linguagem de maquina e linguagem assembly, onde e como colocar as rotinas deste livro, como passar dados entre o basic e as nossas rotinas, e ao final um pequeno programa monitor para aqueles que nao dispoem de um programa assembler.

LINGUAGEM DE MAQUINA E LINGUAGEM ASSEMBLY

Linguagem de maquina e a unica linguagem entendida pelo computador, e e formada por numeros binarios colocados sequencialmente na memoria. Como trabalhar com numeros binarios e complicado e sujeito a erros, utilizamos numeros hexadecimais que tem uma correlacao direta com os numeros binarios, pois cada digito hexadecimal corresponde a 4 digitos binarios. Mesmo assim programar com numeros hexadecimais e dificil, pois estamos acostumados a pensar em termos simbolicos e nao em numeros. Para isso foi desenvolvida a linguagem assembly, que tem uma relacao direta com a linguagem de maquina, mas que se utiliza de simbolos para representar as instrucoes, simbolos estes conhecidos como mneumonicos. Para converter esta linguagem assembly em linguagem de maquina existem programas conhecidos como programas assembler. Caso voce tenha um programa assembler disponivel,podera se utilizar da listagem assembly da rotina, facilitando inclusive o seu entendimento , porem em caso contrario ha uma listagem hexadecimal, ja em linguagem de maquina, que pode ser introduzida diretamente no computador por qualquer monitor simples, como o que esta listado no final deste capitulo. Para facilitar as explicacoes colocamos abaixo a listagem assembly e hexadecimal de uma das rotinas deste livro, a SQRT, que calcula o inteiro da raiz quadrada de um numero de 2 bytes.

LISTAGEM ASSEMBLY

```
4082  C5            SQRT   PUSH BC
4083  D5                   PUSH DE
4084  E5                   PUSH HL
4085  2A 7B 40             LD   HL,(407B)
4088  06 FF                LD   B,FF
408A  11 FF FF             LD   DE,FFFF
408D  04            SQRT1  INC  B
408E  19                   ADD  HL,DE
408F  1B                   DEC  DE
```

```
4090  1B          DEC  DE
4091  38 FA       JR   C SQRTT1
4093  68          LD   L,B
4094  26 00       LD   H,00
4096  22 7B 40    LD   (407B),HL
4099  E1          POP  HL
409A  D1          POP  DE
409B  C1          POP  BC
409C  C9          RET
```

LISTAGEM HEXADECIMAL

```
16.514    C5 D5 E5 2A 7B 40 06 FF
16.522    11 FF FF 04 19 1B 1B 38
16.530    FA 68 26 00 22 7B 40 E1
16.538    D1 C1 C9
```

Podemos verificar que a listagem assembly e composta de varias partes que descreveremos a seguir. A primeira coluna e o endereco onde a rotina sera colocada, e no nosso caso um dado sem muita relevancia, pois todas as rotinas deste livro sao relocaveis, podendo ser colocadas em qualquer regiao da memoria, desde que nao interfira com o basic do computador. Veremos mais tarde quais as regioes da memoria em que poderao ser colocadas as rotinas, sendo importante por enquanto saber que o endereco de inicio colocado tanto na listagem assembly como na listagem hexadecimal (4082h ou 16.514) pode ser alterado. Inclusive todas as rotinas deste livro iniciam com este endereco.

A segunda coluna contem os codigos hexadecimais das instrucoes, sendo a propria linguagem de maquina. Estes codigos hexadecimais podem ter de 1 a 4 bytes, e sao listados sequencialmente na listagem hexadecimal. A terceira coluna e a dos labels, que e um dos fatores que tornam programar em assembler muito mais facil que diretamente em linguagem de maquina. Quando se precisar utilizar um GOTO ou GOSUB em basic, faz-se referencia ao numero da linha, porem em linguagem de maquina e preciso pular para um endereco, e o uso do label torna esta tarefa muito mais facil, sendo o endereco calculado pelo programa assembler na compilacao. Por exemplo, na rotina listada, SQRT1 e um label, e quando ele aparece novamente dentro de uma instrucao, como em JR C SQRT1, o programa assembler calculara o byte a ser colocado na instrucao para se fazer o jump relativo correto, no caso FAh.

A quarta coluna contem o mneumonico da instrucao, que e o que determina o tipo de operacao a ser executada, estando todas as instrucoes do Z-80 listadas na forma resumida no capitulo 1. A quinta coluna contem os operandos envolvidos na instrucao podendo ser registros, pares de registros, numeros (sempre em hexadecimal na listagem assembly, mesmo sem o sufixo h) ou labels.

A listagem hexadecimal e composta de varias linhas com 8 bytes hexadecimais iniciadas pelo endereco decimal. Como ja foi comentado, este endereco nao tem muita significacao, pois todas as rotinas deste livro sao relocaveis.

ONDE COLOCAR AS ROTINAS

Nos microcomputadores com logica SINCLAIR existem varias regioes da memoria que podem ser utilizadas para se colocar rotinas em linguagem de maquina, todas com suas vantagens e desvantagens. Vamos analizar 4 destas regioes, que sao apos o RTP, na area de variaveis, em uma linha REM no final do programa e em linha REM no inicio do programa.

1- APOS O RTP: A variavel do sistema RTP (Ram ToP), enderecos 4004h e 4005h (16.388 e 16.389) contem o endereco do primeiro byte nao disponivel para uso pelo basic, sendo colocado na inicializacao do sistema quando o micro e ligado com o endereco apos o ultimo byte da memoria RAM disponivel. Podemos alterar este valor pokando no RTP o valor desejado do endereco a partir do qual queremos utilizar, seguido do comando NEW. E uma area que poderemos usar sem restricoes porem com a desvantagem de nao ser salva com o comando SAVE. Se quisermos por exemplo colocar uma rotina a partir do endereco 30.000, devemos executar os seguintes comandos no modo direto:

```
POKE 16388,INT (30000/256)
POKE 16389,30000-INT (30000/256)*256
NEW
```

2- NA AREA DE VARIAVEIS: A area de variaveis pode ser utilizada para a colocacao de rotinas em linguagem de maquina, reservando-se espaco para tal atravez do dimensionamento de uma matriz string de uma dimensao, tendo-se o cuidado para que este dimensionamento seja feito apos uma instrucao CLEAR para se assegurar que esta matriz seja realmente o primeiro item na area de variaveis, para que possamos calcular o inicio da rotina. Se precisarmos reservar 50 bytes para uma rotina, podemos faze-lo com os seguintes comandos:

```
CLEAR
DIM A$(50)
```

O endereco de inicio da rotina devera ser calculado dentro do programa toda vez que for feito a chamada atravez de:

```
PEEK 16400+256*PEEK 16401+6
```

A vantagem deste metodo e que a rotina sera salva pelo comando SAVE e nao podera ser listada no video, tendo porem as desvantagens da rotina ser apagada pelos comandos RUN ou CLEAR, e se voce tiver menos que 4K de RAM, a rotina flutuara durante a execucao do programa.

3- EM UMA LINHA REM AO FINAL DO PROGRAMA: Para colocarmos a rotina em uma linha REM ao final de um programa em basic, e necessario primeiro criarmos esta linha. Para conhecermos o endereco do primeiro caractere apos o comando REM podemos utilizar a expressao:
PEEK 16396+256*PEEK 16397-NUMERO DE CARACTERES NA LINHA-1

A principal vantagem deste metodo e que se fizermos um POKE no byte da linha REM que contem a parte menos significativa do numero da linha, com o valor 255, a linha ficara invisivel, nao podendo ser listada, mas continuara a ser salva pelo SAVE. Para calcularmos este byte utilizamos a expressao:
PEEK 16396+256*PEEK 16397-NUMERO DE CARACTERES NA LINHA-6

Uma desvantagem e que a rotina sera deslocada de posicao conforme o programa for alterado, recomendando utilizar-se deste metodo somente com rotinas relocaveis, o que e o caso das rotinas deste livro.

4- EM UMA LINHA REM NO INICIO DO PROGRAMA: E o metodo mais utilizado e conhecido, por que apresenta um inicio dos caracteres conhecido e fixo, 4082h ou 16.514. Para se colocar uma rotina e necessario primeiro criar uma linha REM com um numero suficiente de caracteres, e depois colocar a rotina atravez de qualquer programa monitor. Outra vantagem e que a rotina e salva pelo comando SAVE, e a principal desvantagem e que o byte 76h (118), que e o codigo do NEW-LINE, confundira o programa da ROM que faz aparecer a listagem no video. O byte 7Eh (126) tambem causara alguns efeitos estranhos, pois os proximos 5 bytes nao serao mostrados no video, e poderao desaparecer se a linha for editada.

COMO PASSAR DADOS PARA AS ROTINAS

Passar dados entre o basic e as rotinas em linguagem de maquina e uma tarefa primordial,pois a razao de ser de uma rotina e apresentar resultados, e nao resolvida pelo basic pois o comando USR nao passa nenhum dado para as rotinas chamadas. Existe tambem a complicacao de que as rotinas sao relocaveis, nao havendo dentro delas um endereco fixo para se fazer o POKE dos dados. Para resolvermos este problema utilizamos dois metodos distintos, conforme o numero de bytes a ser passado. Caso o numero de bytes seja igual ou menor a 3, utilizamos 3 bytes na area de variaveis do sistema que nao sao utilizadas pelo sistema operacional. Estes bytes sao, pela ordem em que sao utilizados, os seguintes: 407Bh (16.507), 407Ch (16.508) e 4021h (16.417). Isto significa que se precisarmos passar um byte somente, vamos utilizar o byte 407Bh, se precisarmos passar dois bytes utilizaremos os bytes 407Bh e 407Ch, e caso

precisemos de tres bytes utilizaremos todos os citados. Como ja deve ter sido notado o sufixo h apos o numero significa que este numero esta em hexadecimal, sendo uma excecao a esta regra dentro da listagem assembly onde todos os numeros estarao em hexadecimal. Para passarmos mais de tres bytes para a rotina precisaremos utilizar de um artificio chamado bloco de parametros. Antes de discutirmos sobre bloco de parametros, vamos falar sobre algumas convencoes utilizadas nas descricoes das rotinas. Todo o numero de dois bytes no Z-80 e colocado na memoria com o byte menos significativo primeiro seguido pelo byte mais significativo. Por exemplo, o numero 40A9h colocado nas localizacoes de memoria 407Bh e 407Ch ficaria assim:

407B A9
407C 40

Esta disposicao foi utilizada pelos projetistas do Z-80 para se manter a compatibilidade com o 8080, mais antigo porem com uma extensa biblioteca de software. Utilizaremos para designar o byte menos significativo a abreviacao de LSB, de least significant byte, e para o byte mais significativo MSB, de most significant byte. Estas abreviacoes sao comumentes utilizadas e por isso as utilizaremos.

O bloco de parametros que nos utilizaremos nao e nada mais que uma area da memoria reservada por nos com a quantidade de bytes necessaria, podendo ser utilizada qualquer regiao da memoria, conforme descrito anteriormente, nao precisando ficar proximo a rotina que dela se utilizara. Precisaremos somente informar a rotina o endereco deste bloco de parametros atravez das posicoes 407Bh para o LSB e 407Ch para o MSB do endereco. Como nos conhecemos o endereco do bloco de parametros e facil colocar os valores desejados atravez do uso do POKE. A estrutura do bloco de parametros varia de rotina para rotina e estara descrita na parte CONDICOES DE ENTRADA de cada rotina. Se o bloco de parametros for tambem utilizado para retornar valores para o basic, isto estara especificado nas CONDICOES DE SAIDA de cada rotina.

Em alguns casos havera tambem a necessidade de utilizarmos de um buffer, que nada mais e do que uma area da memoria escolhido por nos, respeitando-se as restricoes ja citadas, com capacidade suficiente para o fim que se destina.

UM PROGRAMA MONITOR

O programa monitor listado a seguir e um pequeno programa em basic para facilitar a entrada dos valores hexadecimais na memoria, nao sendo muito versatil ou completo, devendo ser utilizado somente se nao houver outro programa disponivel. Estes programas podem ser encontrados em revistas especializadas e em varios programas comerciais, facilitando bastante o trabalho de digitacao das rotinas.

```
 10 CLS
 20 PRINT "ENDERECO"
 30 INPUT I
 40 SCROLL
 50 PRINT I;" ";
 60 INPUT X$
 70 IF X$="R" THEN GOTO 10
 80 IF X$="S" THEN STOP
 90 IF LEN X$<>2 THEN GOTO 60
100 PRINT X$
110 LET X=(CODE X$-28)*16+CODE X$(2)-28
120 POKE I,X
130 LET I=I+1
140 GOTO 40
```

O endereco devera ser fornecido em decimal e os codigos em hexadecimal. Para recomecar com novo endereco digite R e para encerrar digite S.

AS ROTINAS

CBSB:Converte um numero binario em string binaria
CBSD:Converte um numero binario em string decimal
CBSH:Converte um numero binario em string hexadecimal
CHKS:Faz o checksum de uma area de memoria
CSBB:Converte string binaria em numero binario
CSDB:Converte string decimal em numero binario
CSHB:Converte string hexadecimal em numero binario
CSLN:Limpa linhas do video
CSTR:Compara strings
DDPO:Divide numero de 16 bits por numero de 8 bits
DDPD:Divide numero de 16 bits por numero de 16 bits
DLBL:Deleta bloco da memoria
EXOR:EXCLUSIVE OR entre dois numeros de 8 bits
INST:Entra com uma string a partir do teclado
MDPD:Multiplica dois numeros de 16 bits
MOBL:Mover bloco de memoria
MOPL:Multiplica dois numeros de 8 bits
MOPO:Multiplica dois numeros de 8 bits rapidamente
MPAD:Adicao em multipla precisao
MPSB:Subtracao em multipla precisao
MSLF:Multiplos deslocamentos a esquerda em 16 bits
MSRG:Multiplos deslocamentos a direita em 16 bits
PRME:Preencher bloco de memoria
PSLN:Preencher linhas do video com caractere desejado
PTST:Imprimir string no video
RVPB:Rodar o video para baixo
RVPC:Rodar o video para cima
RVPD:Rodar o video para a direita
RVPE:Rodar o video para a esquerda
SCDL:Scroll do video para baixo e para a esquerda
SCDR:Scroll do video para baixo e para a direita
SCDW:Scroll do video para baixo
SCLT:Scroll do video para a esquerda
SCRG:Scroll do video para a direita
SCUL:Scroll do video para cima e para a esquerda
SCUP:Scroll do video para cima
SCUR:Scroll do video para cima e para a direita
SQRT:Raiz quadrada de um numero de 16 bits
SSOC:Busca um caractere em uma string
SSTC:Busca dois caracteres em uma string

DESCRICAO

Esta rotina converte um numero binario de 16 bits em uma string binaria com 16 bytes, mais o byte FFh (255) ao final. A string sera formada por bytes 1Ch e 1Dh (28 e 29),o codigo dos caracteres 0 e 1. Como devem ser passados para esta rotina o numero a ser convertido e o endereco onde devera ser colocada a string, 4 bytes no total, utilizamos um bloco de parametros para esta finalidade. Em 407Bh (16.507) e 407Ch (16.508) colocamos o endereco do bloco de parametros, no formato padrao Z-80, ou seja, primeiro o LSB seguido pelo MSB. Os dois primeiros bytes do bloco de parametos serao destinados ao numeros a ser convertido, e os dois ultimos para o endereco onde sera colocado a string binaria, ambos no formato padrao Z-80.

CONDICOES DE ENTRADA

407Bh (16.507) e 407Ch (16.508) devem apontar para um bloco de parametros formado pelo seguinte:

Byte 0:LSB do numero a ser convertido
Byte 1:MSB do numero a ser convertido
Byte 2:LSB do endereco para colocacao da string
Byte 3:MSB do endereco para colocacao da string

A area para colocacao da string devera ter 17 bytes, 16 para o numero e um byte para o terminador FFh (255).

CONDICOES DE SAIDA

A area reservada para a string sera preenchida com bytes 1Ch e 1Dh (28 e 29), conforme o numero binario fornecido, e terminado com o byte FFh (255). O bloco de parametros, 407Bh (16.507) e 407Ch (16.508) nao sao alterados.

LISTAGEM ASSEMBLY

```
4082  C5           CBSB   PUSH BC
4083  D5                  PUSH DE
4084  E5                  PUSH HL
4085  F5                  PUSH AF
4086  2A 7B 40            LD   HL,(407B)
4089  4E                  LD   C,(HL)
408A  23                  INC  HL
408B  46                  LD   B,(HL)
408C  23                  INC  HL
```

```
408D  5E          LD    E,(HL)
408E  23          INC   HL
408F  56          LD    D,(HL)
4090  60          LD    H,B
4091  69          LD    L,C
4092  06 10       LD    B,10
4094  3E 1C CBSB1 LD    A,1C
4096  29          ADD   HL,HL
4097  30 01       JR    NC CBSB2
4099  3C          INC   A
409A  12    CBSB2 LD    (DE),A
409B  13          INC   DE
409C  10 F6       DJNZ .CBSB1
409E  3E FF       LD    A,FF
40A0  12          LD    (DE),A
40A1  F1          POP   AF
40A2  E1          POP   HL
40A3  D1          POP   DE
40A4  C1          POP   BC
40A5  C9          RET
```

LISTAGEM HEXADECIMAL

```
16.514    C5 D5 E5 F5 2A 7B 40 4E
16.522    23 46 23 5E 23 56 60 69
16.530    06 10 3E 1C 29 30 01 3C
16.538    12 13 10 F6 3E FF 12 F1
16.546    E1 D1 C1 C9
```

DESCRICAO

Esta rotina converte um numero binario de 16 bits em uma string decimal com 5 bytes, mais o byte FFh (255) ao final. A string sera formada por bytes 1Ch a 25h (28 a 37),o codigo dos caracteres 0 a 9. Como devem ser passados para esta rotina o numero a ser convertido e o endereco onde devera ser colocada a string, 4 bytes no total, utilizamos um bloco de parametros para esta finalidade. Em 407Bh (16.507) e 407Ch (16.508) colocamos o endereco do bloco de parametros, no formato padrao Z-80, ou seja, primeiro o LSB seguido pelo MSB. Os dois primeiros bytes do bloco de parametos serao destinados ao numeros a ser convertido, e os dois ultimos para o endereco onde sera colocado a string binaria, ambos no formato padrao Z-80.

CONDICOES DE ENTRADA

407Bh (16.507) e 407Ch (16.508) devem apontar para um bloco de parametros formado pelo seguinte:

Byte 0:LSB do numero a ser convertido
Byte 1:MSB do numero a ser convertido
Byte 2:LSB do endereco para colocacao da string
Byte 3:MSB do endereco para colocacao da string

A area para colocacao da string devera ter 6 bytes, 5 para o numero e um byte para o terminador FFh (255).

CONDICOES DE SAIDA

A area reservada para a string sera preenchida com bytes 1Ch a 25h (28 a 37), conforme o numero binario fornecido, e terminado com o byte FFh (255). O bloco de parametros, 407Bh (16.507) e 407Ch (16.508) nao sao alterados.

LISTAGEM ASSEMBLY

```
4082  C5            CBSD   PUSH BC
4083  D5                   PUSH DE
4084  E5                   PUSH HL
4085  F5                   PUSH AF
4086  2A 7B 40             LD   HL,(407B)
4089  4E                   LD   C,(HL)
408A  23                   INC  HL
408B  46                   LD   B,(HL)
408C  23                   INC  HL
```

```
408D  5E                      LD    E,(HL)
408E  23                      INC   HL
408F  56                      LD    D,(HL)
4090  60                      LD    H,B
4091  69                      LD    L,C
4092  01 F0 D8                LD    BC,D8F0
4095  18 1C                   JR    CBSD6
4097  01 18 FC        CBSD1   LD    BC,FC18
409A  18 17                   JR    CBSD6
409C  01 9C FF        CBSD2   LD    BC,FF9C
409F  18 12                   JR    CBSD6
40A1  01 F6 FF        CBSD3   LD    BC,FFF6
40A4  18 0D                   JR    CBSD6
40A6  01 FF FF        CBSD4   LD    BC,FFFF
40A9  18 08                   JR    CBSD6
40AB  3E FF           CBSD5   LD    A,FF
40AD  12                      LD    (DE),A
40AE  F1                      POP   AF
40AF  E1                      POP   HL
40B0  D1                      POP   DE
40B1  C1                      POP   BC
40B2  C9                      RET
40B3  3E FF           CBSD6   LD    A,FF
40B5  3C              CBSD7   INC   A
40B6  09                      ADD   HL,BC
40B7  38 FC                   JR    C CBSD7
40B9  B7                      OR    A
40BA  ED 42                   SBC   HL,BC
40BC  C6 1C                   ADD   A,1C
40BE  12                      LD    (DE),A
40BF  13                      INC   DE
40C0  79                      LD    A,C
40C1  FE F0                   CP    F0
40C3  28 D2                   JR    Z CBSD1
40C5  FE 18                   CP    18
40C7  28 D3                   JR    Z CBSD2
40C9  FE 9C                   CP    9C
40CB  28 D4                   JR    Z CBSD3
40CD  FE F6                   CP    F6
40CF  28 D5                   JR    Z CBSD4
40D1  18 D8                   JR    CBSD5
```

LISTAGEM HEXADECIMAL

```
16.514      C5 D5 E5 F5 2A 7B 40 4E
16.522      23 46 23 5E 23 56 60 69
16.530      01 F0 D8 18 1C 01 18 FC
16.538      18 17 01 9C FF 18 12 01
16.546      F6 FF 18 0D 01 FF FF 18
16.554      08 3E FF 12 F1 E1 D1 C1
16.562      C9 3E FF 3C 09 38 FC B7
16.570      ED 42 C6 1C 12 13 79 FE
16.578      F0 28 D2 FE 18 28 D3 FE
```

```
16.586      9C 28 D4 FE F6 28 D5 18
16.594      D8
```

DESCRICAO

Esta rotina converte um numero binario de 16 bits em uma string hexadecimal com 4 bytes, mais o byte FFh (255) ao final. A string sera formada por bytes 1Ch a 2Bh (28 a 43), o codigo dos caracteres 0 a 9 e A a F. Como devem ser passados para esta rotina o numero a ser convertido e o endereco onde devera ser colocada a string, 4 bytes no total, utilizamos um bloco de parametros para esta finalidade. Em 407Bh (16.507) e 407Ch (16.508) colocamos o endereco do bloco de parametros, no formato padrao Z-80, ou seja, primeiro o LSB seguido pelo MSB. Os dois primeiros bytes do bloco de parametros serao destinados ao numero a ser convertido, e os dois ultimos para o endereco onde sera colocado a string hexadecimal, ambos no formato padrao Z-80.

CONDICOES DE ENTRADA

407Bh (16.507) e 407Ch (16.508) devem apontar para um bloco de parametros formado pelo seguinte:

Byte 0:LSB do numero a ser convertido
Byte 1:MSB do numero a ser convertido
Byte 2:LSB do endereco para colocacao da string
Byte 3:MSB do endereco para colocacao da string

A area para colocacao da string devera ter 5 bytes, 4 para o numero e um byte para o terminador FFh (255).

CONDICOES DE SAIDA

A area reservada para a string sera preenchida com bytes 1Ch a 2Bh (28 a 43), conforme o numero binario fornecido, e terminado com o byte FFh (255). O bloco de parametros, 407Bh (16.507) e 407Ch (16.508) nao sao alterados.

LISTAGEM ASSEMBLY

```
4082  C5            CBSH  PUSH BC
4083  D5                  PUSH DE
4084  E5                  PUSH HL
4085  F5                  PUSH AF
4086  2A 7B 40            LD   HL,(407B)
4089  4E                  LD   C,(HL)
408A  23                  INC  HL
408B  46                  LD   B,(HL)
```

```
408C  23                INC  HL
408D  5E                LD   E,(HL)
408E  23                INC  HL
408F  56                LD   D,(HL)
4090  60                LD   H,B
4091  69                LD   L,C
4092  06 04             LD   B,04
4094  AF         CBSH1  XOR  A
4095  29                ADD  HL,HL
4096  17                RLA
4097  29                ADD  HL,HL
4098  17                RLA
4099  29                ADD  HL,HL
409A  17                RLA
409B  29                ADD  HL,HL
409C  17                RLA
409D  C6 1C             ADD  A,1C
409F  12                LD   (DE),A
40A0  13                INC  DE
40A1  10 F1             DJNZ CBSH1
40A3  3E FF             LD   A,FF
40A5  12                LD   (DE),A
40A6  F1                POP  AF
40A7  E1                POP  HL
40A8  D1                POP  DE
40A9  C1                POP  BC
40AA  C9                RET
```

LISTAGEM HEXADECIMAL

```
16.514     C5 D5 E5 F5 2A 7B 40 4E
16.522     23 46 23 5E 23 56 60 69
16.530     06 04 AF 29 17 29 17 29
16.538     17 29 17 C6 1C 12 13 10
16.546     F1 3E FF 12 F1 E1 D1 C1
16.554     C9
```

DESCRICAO

Esta rotina executa o checksum de um bloco de memoria, podendo ser utilizado para verificar se os dados em um determinado bloco nao estao adulterados. Este checksum e calculado na forma aditiva, sendo todos os bytes do bloco somados no acumulador do Z-80. Como temos de fornecer 4 bytes para a rotina, o endereco do bloco e o numero de bytes, e teremos um byte como resultado, utilizamos um bloco de parametros para esta finalidade. Ate 64K de memoria podem ser checados com esta rotina.

CONDICOES DE ENTRADA

Devemos colocar em 407Bh (16.507) e 407Ch (16.508) o endereco do bloco de parametros, no formato padrao Z-80, primeiro o LSB seguido pelo MSB. O bloco de parametros sera formado pelo seguinte:

Byte 0:LSB do inicio do bloco de memoria
Byte 1:MSB do inicio do bloco de memoria
Byte 2:LSB do numero de bytes no bloco de memoria
Byte 3:MSB do numero de bytes no bloco de memoria
Byte 4:Reservado para o resultado

CONDICOES DE SAIDA

O byte 4 do bloco de parametros contem o checksum do bloco de memoria. 407Bh (16.507), 407Ch (16.508) e os primeiros quatro bytes do bloco de parametros nao sao alterados.

LISTAGEM ASSEMBLY

```
4082  C5          CHKS   PUSH BC
4083  D5                 PUSH DE
4084  E5                 PUSH HL
4085  F5                 PUSH AF
4086  2A 7B 40           LD   HL,(407B)
4089  4E                 LD   C,(HL)
408A  23                 INC  HL
408B  46                 LD   B,(HL)
408C  23                 INC  HL
408D  5E                 LD   E,(HL)
408E  23                 INC  HL
408F  56                 LD   D,(HL)
4090  E5                 PUSH HL
4091  60                 LD   H,B
```

```
4092  69                      LD    L,C
4093  01 01 00                LD    BC,0001
4096  AF                      XOR   A
4097  86             CHKS1    ADD   A,(HL)
4098  23                      INC   HL
4099  B7                      OR    A
409A  EB                      EX    DE,HL
409B  ED 42                   SBC   HL,BC
409D  EB                      EX    DE,HL
409E  20 F7                   JR    NZ CHKS1
40A0  E1                      POP   HL
40A1  23                      INC   HL
40A2  77                      LD    (HL),A
40A3  F1                      POP   AF
40A4  E1                      POP   HL
40A5  D1                      POP   DE
40A6  C1                      POP   BC
40A7  C9                      RET
```

LISTAGEM HEXADECIMAL

```
16.514     C5 D5 E5 F5 2A 7B 40 4E
16.522     23 46 23 5E 23 56 E5 60
16.530     69 01 01 00 AF 86 23 B7
16.538     EB ED 42 EB 20 F7 E1 23
16.546     77 F1 E1 D1 C1 C9
```

DESCRICAO

Esta rotina converte uma string binaria em um numero binario de 16 bits. A string binaria deve ser formada por bytes 1Ch e 1Dh (28 e 29), os codigos utilizados pelos micros com logica Sinclair para os numeros de 0 e 1. A string pode ter qualquer numero de bytes, porem somente os 16 ultimos serao considerados, e deve ser terminada pelo byte FFh (255). Nao e feito qualquer vericacao se os bytes da string estao dentro da faixa permitida, sendo que a rotina mostrara um resultado errado se isto ocorrer. O endereco da string devera ser colocado nas posicoes 407Bh (16.507) e 407Ch (16.508), no formato padrao Z-80, primeiro o LSB e depois o MSB. O resultado sera colocado nas posicoes 407Bh (16.507) e 407Ch (16.508), tambem no formato padrao Z-80.

CONDICOES DE ENTRADA

407Bh (16.507) e 407Ch (16.508) devem apontar para a area da memoria onde esteja colocada a string binaria.

CONDICOES DE SAIDA

407Bh (16.507) e 407Ch (16.508) contem o resultado na faixa de 0000h a FFFFh (0 a 65.535). A area da memoria com a string binaria nao e alterada.

LISTAGEM ASSEMBLY

```
4082  C5                CSBB   PUSH BC
4083  D5                       PUSH DE
4084  E5                       PUSH HL
4085  F5                       PUSH AF
4086  ED 5B 7B 40              LD   DE,(407B)
408A  21 00 00                 LD   HL,0000
408D  06 00                    LD   B,00
408F  1A                CSBB1  LD   A,(DE)
4090  FE FF                    CP   FF
4092  28 08                    JR   Z CSBB2
4094  D6 1C                    SUB  1C
4096  4F                       LD   C,A
4097  29                       ADD  HL,HL
4098  09                       ADD  HL,BC
4099  13                       INC  DE
409A  18 F3                    JR   CSBB1
409C  22 7B 40          CSBB2  LD   (407B),HL
409F  F1                       POP  AF
```

```
40A0  E1                    POP  HL
40A1  D1                    POP  DE
40A2  C1                    POP  BC
40A3  C9                    RET
```

LISTAGEM HEXADECIMAL

```
16.514    C5 D5 E5 F5 ED 5B 7B 40
16.522    21 00 00 06 00 1A FE FF
16.530    28 08 D6 1C 4F 29 09 13
16.538    18 F3 22 7B 40 F1 E1 D1
16.546    C1 C9
```

DESCRICAO

Esta rotina converte uma string decimal em um numero binario de 16 bits. A string decimal deve ser formada por bytes 1Ch a 25h (28 a 37), os codigos utilizados pelos micros com logica Sinclair para os numeros de 0 a 9. A string pode ter de 0 a 5 bytes, e deve ser terminada pelo byte FFh (255). Nao e feito qualquer vericacao se os bytes da string estao dentro da faixa permitida, sendo que a rotina mostrara um resultado errado se isto ocorrer, ou se houver mais de 5 bytes na string, fora o terminador FFh, ou se o valor da string for maior que 65.535. O endereco da string devera ser colocado nas posicoes 407Bh (16.507) e 407Ch (16.508), no formato padrao Z-80, primeiro o LSB e depois o MSB. O resultado sera colocado nas posicoes 407Bh (16.507) e 407Ch (16.508), tambem no formato padrao Z-80.

CONDICOES DE ENTRADA

407Bh (16.507) e 407Ch (16.508) devem apontar para a area da memoria onde esteja colocada a string decimal.

CONDICOES DE SAIDA

407Bh (16.507) e 407Ch (16.508) contem o resultado na faixa de 0000h a FFFFh (0 a 65.535). A area da memoria com a string decimal nao e alterada.

LISTAGEM ASSEMBLY

```
4082  C5            CSDB    PUSH BC
4083  D5                    PUSH DE
4084  E5                    PUSH HL
4085  F5                    PUSH AF
4086  ED 5B 7B 40           LD   DE,(407B)
408A  21 00 00              LD   HL,0000
408D  1A            CSDB1   LD   A,(DE)
408E  FE FF                 CP   FF
4090  28 0F                 JR   Z CSDB2
4092  29                    ADD  HL,HL
4093  E5                    PUSH HL
4094  29                    ADD  HL,HL
4095  29                    ADD  HL,HL
4096  C1                    POP  BC
4097  09                    ADD  HL,BC
4098  D6 1C                 SUB  1C
```

```
409A   4F                      LD    C,A
409B   06 00                   LD    B,00
409D   09                      ADD   HL,BC
409E   13                      INC   DE
409F   18 EC                   JR    CSDB1
40A1   22 7B 40       CSDB2    LD    (407B),HL
40A4   F1                      POP   AF
40A5   E1                      POP   HL
40A6   D1                      POP   DE
40A7   C1                      POP   BC
40A8   C9                      RET
```

LISTAGEM HEXADECIMAL

```
16.514      C5 D5 E5 F5 ED 5B 7B 40
16.522      21 00 00 1A FE FF 28 0F
16.530      29 E5 29 29 C1 09 D6 1C
16.538      4F 06 00 09 13 18 EC 22
16.546      7B 40 F1 E1 D1 C1 C9
```

DESCRICAO

Esta rotina converte uma string hexadecimal em um numero binario de 16 bits. A string hexadecimal deve ser formada por bytes 1Ch a 2Bh (28 a 43), os codigos utilizados pelos micros com logica Sinclair para os numeros de 0 a 9 e as letras de A a F. A string pode ter qualquer numero de bytes, porem somente os quatro ultimos serao considerados, e deve ser terminada pelo byte FFh (255). Nao e feito qualquer verificacao se os bytes da string estao dentro da faixa permitida, sendo que a rotina mostrara um resultado errado se isto ocorrer. O endereco da string devera ser colocado nas posicoes 407Bh (16.507) e 407Ch (16.508), no formato padrao Z-80, primeiro o LSB e depois o MSB. O resultado sera colocado nas posicoes 407Bh (16.507) e 407Ch (16.508), tambem no formato padrao Z-80.

CONDICOES DE ENTRADA

407Bh (16.507) e 407Ch (16.508) devem apontar para a area da memoria onde esteja colocada a string hexadecimal.

CONDICOES DE SAIDA

407Bh (16.507) e 407Ch (16.508) contem o resultado na faixa de 0000h a FFFFh (0 a 65.535).

LISTAGEM ASSEMBLY

```
4082  C5               CSHB    PUSH BC
4083  D5                       PUSH DE
4084  E5                       PUSH HL
4085  F5                       PUSH AF
4086  ED 5B 7B 40              LD   DE,(407B)
408A  21 00 00                 LD   HL,0000
408D  06 00                    LD   B,00
408F  1A               CSHB1   LD   A,(DE)
4090  FE FF                    CP   FF
4092  28 0B                    JR   Z CSHB2
4094  29                       ADD  HL,HL
4095  29                       ADD  HL,HL
4096  29                       ADD  HL,HL
4097  29                       ADD  HL,HL
4098  D6 1C                    SUB  1C
409A  4F                       LD   C,A
409B  09                       ADD  HL,BC
```

```
409C  13                  INC  DE
409D  18 F0               JR   CSHB1
409F  22 7B 40    CSHB2   LD   (407B),HL
40A2  F1                  POP  AF
40A3  E1                  POP  HL
40A4  D1                  POP  DE
40A5  C1                  POP  BC
40A6  C9                  RET
```

LISTAGEM HEXADECIMAL

```
16.514    C5 D5 E5 F5 ED 5B 7B 40
16.522    21 00 00 06 00 1A FE FF
16.530    28 0B 29 29 29 29 D6 1C
16.538    4F 09 13 18 F0 22 7B 40
16.546    F1 E1 D1 C1 C9
```

DESCRICAO

Esta rotina limpa linhas do video, devendo ser especificado a linha inicial e a linha final. Nao e verificado se os numeros de linha fornecidos estao dentro da faixa permitida, que e de 00h a 17h (0 a 23), e se o numero final e maior que o inicial. Se estas condicoes nao forem seguidas o computador ficara fora de controle, perdendo-se todos os programas da memoria.

CONDICOES DE ENTRADA

O numero da linha inicial devera ser colocado em 407Bh (16.507) e o numero da linha final em 407Ch (16.508).

CONDICOES DE SAIDA

As linhas do video especificadas serao limpas. 407Bh (16.507) e 407Ch (16.508) nao sao alterados.

LISTAGEM ASSEMBLY

```
4082  C5          CSLN   PUSH BC
4083  D5                 PUSH DE
4084  E5                 PUSH HL
4085  F5                 PUSH AF
4086  2A 7B 40           LD   HL,(407B)
4089  E5                 PUSH HL
408A  7C                 LD   A,H
408B  95                 SUB  L
408C  3C                 INC  A
408D  6F                 LD   L,A
408E  26 00              LD   H,00
4090  E5                 PUSH HL
4091  29                 ADD  HL,HL
4092  29                 ADD  HL,HL
4093  29                 ADD  HL,HL
4094  29                 ADD  HL,HL
4095  29                 ADD  HL,HL
4096  C1                 POP  BC
4097  09                 ADD  HL,BC
4098  EB                 EX   HL,BC
4099  E1                 POP  HL
409A  26 00              LD   H,00
409C  E5                 PUSH HL
409D  29                 ADD  HL,HL
409E  29                 ADD  HL,HL
```

```
409F  29                      ADD  HL,HL
40A0  29                      ADD  HL,HL
40A1  29                      ADD  HL,HL
40A2  C1                      POP  BC
40A3  09                      ADD  HL,BC
40A4  44                      LD   B,H
40A5  4D                      LD   C,L
40A6  2A 0C 40                LD   HL,(400C)
40A9  09                      ADD  HL,BC
40AA  23             CSLN1    INC  HL
40AB  7E                      LD   A,(HL)
40AC  FE 76                   CP   76
40AE  28 02                   JR   Z CSLN2
40B0  AF                      XOR  A
40B1  77                      LD   (HL),A
40B2  1B             CSLN2    DEC  DE
40B3  7A                      LD   A,D
40B4  B3                      OR   E
40B5  20 F3                   JR   NZ CSLN1
40B7  F1                      POP  AF
40B8  E1                      POP  HL
40B9  D1                      POP  DE
40BA  C1                      POP  BC
40BB  C9                      RET
```

LISTAGEM HEXADECIMAL

```
16.514     C5 D5 E5 F5 2A 7B 40 E5
16.522     7C 95 3C 6F 26 00 E5 29
16.530     29 29 29 29 C1 09 EB E1
16.538     26 00 E5 29 29 29 29 29
16.546     C1 09 44 4D 2A 0C 40 09
16.554     23 7E FE 76 28 02 AF 77
16.562     1B 7A B3 20 F3 F1 E1 D1
16.570     C1 C9
```

DESCRICAO

Esta rotina compara duas strings, sendo que o significado de string neste caso difere fundamentalmente do conceito de string do basic. String para esta rotina significa qualquer bloco de memoria, podendo ser dados numericos, alfanumericos ou ate mesmo programas. As strings podem ter um comprimento maximo 255 bytes. Como devem ser passados para esta rotina o endereco das duas strings e os seus comprimentos, e mais um byte para o resultado, utilizamos para esta finalidade de um bloco de parametros.

CONDICOES DE ENTRADA

Devemos colocar em 407Bh (16.507) e em 407Ch (16.508) o endereco do bloco de parametros, no formato padrao Z-80, primeiro o LSB seguido pelo MSB. O bloco de parametros sera formado pelo seguinte:

Byte 0:Numero de bytes da string 1
Byte 1:LSB do endereco da string 1
Byte 2:MSB do endereco da string 2
Byte 3:Numero de bytes da string 2
Byte 4:LSB do endereco da string 2
Byte 5:MSB do endereco da string 2
Byte 6:Reservado para o resultado

CONDICOES DE SAIDA

No byte 6 do bloco de parametros sera colocado 00h (0) se as duas strings forem iguais, 01h (1) se a string 1 for maior que a string 2, e FFh (255) se a string 1 for menor que a string 2. Se o comprimento das duas strings forem diferentes, e uma string for uma substring da outra, sera colocado 01h (1) para a string 1 maior que a string 2, e FFh (255) para a string 1 menor que a string 2.

LISTAGEM ASSEMBLY

```
4082  C5          CSTR   PUSH BC
4083  D5                 PUSH DE
4084  E5                 PUSH HL
4085  F5                 PUSH AF
4086  2A 7B 40           LD   HL,(407B)
4089  46                 LD   B,(HL)
408A  0E 00              LD   C,00
408C  7E                 LD   A,(HL)
408D  23                 INC  HL
```

```
408E  5E                LD    E,(HL)
408F  23                INC   HL
4090  56                LD    D,(HL)
4091  23                INC   HL
4092  BE                CP    (HL)
4093  28 09             JR    Z CSTR2
4095  38 05             JR    C CSTR1
4097  46                LD    B,(HL)
4098  0E 01             LD    C,01
409A  18 02             JR    CSTR2
409C  0E FF      CSTR1  LD    C,FF
409E  D5         CSTR2  PUSH  DE
409F  23                INC   HL
40A0  5E                LD    E,(HL)
40A1  23                INC   HL
40A2  56                LD    D,(HL)
40A3  E3                EX    (SP),HL
40A4  EB                EX    DE,HL
40A5  1A         CSTR3  LD    A,(DE)
40A6  96                SUB   (HL)
40A7  20 07             JR    NZ CSTR4
40A9  23                INC   HL
40AA  13                INC   DE
40AB  10 F8             DJNZ  CSTR3
40AD  79                LD    A,C
40AE  18 06             JR    CSTR5
40B0  3E 01      CSTR4  LD    A,01
40B2  30 02             JR    NC CSTR5
40B4  3E FF             LD    A,FF
40B6  E1         CSTR5  POP   HL
40B7  23                INC   HL
40B8  77                LD    A,(HL)
40B9  F1                POP   AF
40BA  E1                POP   HL
40BB  D1                POP   DE
40BC  C1                POP   BC
40BD  C9                RET
```

LISTAGEM HEXADECIMAL

```
16.514    C5 D5 E5 F5 2A 7B 40 46
16.522    0E 00 7E 23 5E 23 56 23
16.530    BE 28 09 38 05 46 0E 01
16.538    18 02 0E FF D5 23 5E 23
16.546    56 E3 EB 1A 96 20 07 23
16.554    13 10 F8 79 18 06 3E 01
16.562    30 02 3E FF E1 23 77 F1
16.570    E1 D1 C1 C9
```

DESCRICAO

Esta rotina divide um numero de 16 bits por outro numero de 8 bits. O dividendo de 16 bits devera estar nas posicoes 407Bh (16.507) e 407Ch (16.508), no formato padrao Z-80, primeiro o LSB seguido pelo MSB. O divisor de 8 bits devera estar na posicao 4021h (16.417). O resultado estara em 407Bh (16.507) e 407Ch (16.508) no formato padrao Z-80, e o resto estara em 4021h (16.417). Divisao por zero resultara em FFFFh (65.535), o que e incorreto.

CONDICOES DE ENTRADA

O dividendo de 16 bits devera estar em 407Bh (16.507) e 407Ch (16.508) no formato padrao Z-80, e o divisor de 8 bits devera estar em 4021h (16.417).

CONDICOES DE SAIDA

O quociente de 16 bits estara em 407Bh (16.507) e 407Ch (16.508) no formato padrao Z-80, e o resto de 8 bits estara em 4021h (16.417).

LISTAGEM ASSEMBLY

```
4082  C5         DDPO   PUSH BC
4083  E5                PUSH HL
4084  F5                PUSH AF
4085  2A 7B 40          LD   HL,(407B)
4088  3A 21 40          LD   A,(4021)
408B  4F                LD   C,A
408C  06 10             LD   B,10
408E  AF                XOR  A
408F  29         DDPO1  ADD  HL,HL
4090  8F                ADC  A,A
4091  2C                INC  L
4092  91                SUB  C
4093  30 02             JR   NC DDPO2
4095  81                ADD  A,C
4096  2D                DEC  L
4097  10 F6      DDPO2  DJNZ DDPO1
4099  22 7B 40          LD   (407B),HL
409C  32 21 40          LD   (4021),A
409F  F1                POP  AF
40A0  E1                POP  HL
40A1  C1                POP  BC
40A2  C9                RET
```

LISTAGEM HEXADECIMAL

```
16.514    C5 E5 F5 2A 7B 40 3A 21
16.522    40 4F 06 10 AF 29 8F 2C
16.530    91 30 02 81 2D 10 F6 22
16.538    7B 40 32 21 40 F1 E1 C1
16.546    C9
```

DESCRICAO

Esta rotina divide um numero de 16 bits por outro numero de 16 bits. Como devem ser passados para esta rotina dois numeros de 2 bytes,o dividendo e o divisor, e ela deve retornar outros dois numeros de 2 bytes,o quociente e o resto, sera utilizado um bloco de parametros com esta finalidade. Divisao por zero acarretara um resultado incorreto de FFFFh (65.535).

CONDICOES DE ENTRADA

407Bh (16.507) e 407Ch (16.508) devem conter o endereco do bloco de parametros no formato padrao Z-80, primeiro o LSB seguido do MSB. O bloco de parametros sera formado pelo seguinte:

Byte 0:LSB do dividendo
Byte 1:MSB do dividendo
Byte 2:LSB do divisor
Byte 3:MSB do divisor
Byte 4:Reservado para o LSB do quociente
Byte 5:Reservado para o MSB do quociente
Byte 6:Reservado para o LSB do resto
Byte 7:Reservado para o MSB do resto

CODICOES DE SAIDA

O quociente de 16 bits estara nos bytes 4 e 5 do bloco de parametros, no formato padrao Z-80, e o resto estara nos bytes 6 e 7, tambem no formato padrao Z-80. As posicoes 407Bh (16.507), 407Ch (16.508) e os primeiros quatros bytes do bloco de parametros nao sao alterados.

LISTAGEM ASSEMBLY

```
4082  C5           DDPD   PUSH BC
4083  D5                  PUSH DE
4084  E5                  PUSH HL
4085  F5                  PUSH AF
4086  2A 7B 40            LD   HL,(407B)
4089  5E                  LD   E,(HL)
408A  23                  INC  HL
408B  56                  LD   D,(HL)
408C  23                  INC  HL
408D  4E                  LD   C,(HL)
408E  23                  INC  HL
408F  46                  LD   B,(HL)
```

```
4090  E5                 PUSH HL
4091  21 00 00           LD   HL,0000
4094  3E 10              LD   A,10
4096  EB          DDPD1  EX   DE,HL
4097  29                 ADD  HL,HL
4098  EB                 EX   DE,HL
4099  ED 6A              ADC  HL,HL
409B  13                 INC  DE
409C  B7                 OR   A
409D  ED 42              SBC  HL,BC
409F  30 02              JR   NC DDPD2
40A1  1B                 DEC  DE
40A2  09                 ADD  HL,BC
40A3  3D          DDPD2  DEC  A
40A4  20 F0              JR   NZ DDPD1
40A6  E3                 EX   (SP),HL
40A7  23                 INC  HL
40A8  73                 LD   (HL),E
40A9  23                 INC  HL
40AA  72                 LD   (HL),D
40AB  D1                 POP  DE
40AC  23                 INC  HL
40AD  73                 LD   (HL),E
40AE  23                 INC  HL
40AF  72                 LD   (HL),D
40B0  F1                 POP  AF
40B1  E1                 POP  HL
40B2  D1                 POP  DE
40B3  C1                 POP  BC
40B4  C9                 RET
```

LISTAGEM HEXADECIMAL

```
16.514    C5 D5 E5 F5 2A 7B 40 5E
16.522    23 56 23 4E 23 46 E5 21
16.530    00 00 3E 10 EB 29 EB ED
16.538    6A 13 B7 ED 42 30 02 1B
16.546    09 3D 20 F0 E3 23 73 23
16.554    72 D1 23 73 23 72 F1 E1
16.562    D1 C1 C9
```

DESCRICAO

Esta rotina deleta um bloco de memoria dentro de um bloco maior, relocando-se a parte posterior do bloco maior para o final da parte anterior deste bloco. Uma das utilizacoes desta rotina poderia ser em processamento de texto, para deletar uma parte do texto e unir as partes restantes. Como devemos passar para esta rotina o endereco do bloco a ser deletado e do bloco maior, e tambem o comprimento dos dois blocos utilizamos para esta finalidade um bloco de parametros.

CONDICOES DE ENTRADA

Devemos colocar em 407Bh (16.507) e em 407Ch (16.508) o endereco do bloco de parametros, no formato padrao Z-80, primeiro o LSB seguido do MSB. O bloco de parametros deve ser formado pelo seguinte:

Byte 0:LSB do endereco do bloco a ser deletado
Byte 1:MSB do endereco do bloco a ser deletado
Byte 2:LSB do # bytes do bloco a ser deletado
Byte 3:MSB do # bytes do bloco a ser deletado
Byte 4:LSB do endereco do bloco maior
Byte 5:MSB do endereco do bloco maior
Byte 6:LSB do # bytes do bloco maior
Byte 7:MSB do # bytes do bloco maior

CONDICOES DE SAIDA

A parte posterior do bloco maior estara sobre o bloco a ser deletado, formando um unico bloco com a parte anterior deste mesmo bloco. Ao final do novo bloco havera um lixo, ou seja, a memoria estara preenchida com dados que nao pertencem ao novo bloco. O bloco de parametros, 407Bh (16.507) e 407Ch (16.508) nao sao alterados.

LISTAGEM ASSEMBLY

```
4082  C5          DLBL   PUSH BC
4083  D5                 PUSH DE
4084  E5                 PUSH HL
4085  F5                 PUSH AF
4086  2A 7B 40           LD   HL,(407B)
4089  5E                 LD   E,(HL)
408A  23                 INC  HL
408B  56                 LD   D,(HL)
408C  D5                 PUSH DE
```

```
408D  23         INC  HL
408E  4E         LD   C,(HL)
408F  23         INC  HL
4090  46         LD   B,(HL)
4091  EB         EX   DE,HL
4092  09         ADD  HL,BC
4093  EB         EX   DE,HL
4094  D5         PUSH DE
4095  23         INC  HL
4096  5E         LD   E,(HL)
4097  23         INC  HL
4098  56         LD   D,(HL)
4099  23         INC  HL
409A  4E         LD   C,(HL)
409B  23         INC  HL
409C  46         LD   B,(HL)
409D  EB         EX   DE,HL
409E  09         ADD  HL,BC
409F  D1         POP  DE
40A0  B7         OR   A
40A1  ED 52      SBC  HL,DE
40A3  44         LD   B,H
40A4  4D         LD   C,L
40A5  E1         POP  HL
40A6  EB         EX   DE,HL
40A7  ED B0      LDIR
40A9  F1         POP  AF
40AA  E1         POP  HL
40AB  D1         POP  DE
40AC  C1         POP  BC
40AD  C9         RET
```

LISTAGEM HEXADECIMAL

```
16.514    C5 D5 E5 F5 2A 7B 40 5E
16.522    23 56 D5 23 4E 23 46 EB
16.530    09 EB D5 23 5E 23 56 23
16.538    4E 23 46 EB 09 D1 B7 ED
16.546    52 44 4D E1 EB ED B0 F1
16.554    E1 D1 C1 C9
```

DESCRICAO

Esta rotina faz a operacao EXCLUSIVE OR entre dois operandos de 8 bits. A operacao e executada bit a bit, sendo o resultado igual a 1 somente se os bits dos dois operandos forem diferentes. Esta operecao nao existe no basic.

CONDICOES DE ENTRADA

Os dois operados devem estar em 407Bh (16.507) e em 407Ch (16.508).

CONDICOES DE SAIDA

O resultado da operacao EXCLUSIVE OR estara em 407Bh (16.507) e 407Ch (16.508) estara zerado.

LISTAGEM ASSEMBLY

```
4082  E5          EXOR   PUSH HL
4083  F5                 PUSH AF
4084  2A 7B 40           LD   HL,(407B)
4087  7C                 LD   A,H
4088  AD                 XOR  L
4089  6F                 LD   L,A
408A  26 00              LD   H,00
408C  22 7B 40           LD   (407B),HL
408F  F1                 POP  AF
4090  E1                 POP  HL
4091  C9                 RET
```

LISTAGEM HEXADECIMAL

```
16.514     E5 F5 2A 7B 40 7C AD 6F
16.522     26 00 22 7B 40 F1 E1 C9
```

DESCRICAO

Esta rotina entra com uma string a partir do teclado. Deve ser especificado os numeros de linha e coluna para impressao dos caracteres que estao sendo digitados, porem deve se ter o cuidado para que a string nao passe para outra linha do video, limitando-se assim o maximo comprimento da string em 32 caracteres. Os caracteres de comando reconhecidos por esta rotina sao o NEW-LINE (ou ENTER) para se encerrar a entrada de dados e o RUBOUT (ou DELETE) para se apagar o caractere a esquerda do cursor, que e um caractere grafico piscante. Deve ser especificado tambem o numero maximo de caracteres a serem digitados, e o endereco do buffer que vai receber estes caracteres. Este buffer deve ter um byte a mais que o comprimento maximo especificado para receber o byte FFh (255) ao final. Se a entrada da string e encerrada com um numero de bytes menor que o maximo, o byte FFh (255) e colocado em seguida, ficando o resto do buffer sem significacao.

CONDICOES DE ENTRADA

Devemos colocar em 407Bh (16.507) e em 407Ch (16.508) o endereco do bloco de parametros, no formato padrao Z-80, primeiro o LSB seguido pelo MSB. O bloco de parametros deve ser formado pelo seguinte:

Byte 0:Numero da linha a ser impresso no video
Byte 1:Numero da coluna a ser impresso no video
Byte 2:Numero de bytes maximo da string
Byte 3:LSB do endereco do buffer
Byte 4:MSB do endereco do buffer

CONDICOES DE SAIDA

O buffer e preenchido com os caracteres digitados, finalizado pelo byte FFh (255). O bloco de parametros, 407Bh (16.507) e 407Ch (16.508) nao sao alterados.

LISTAGEM ASSEMBLY

```
4082  C5          INST   PUSH BC
4083  D5                 PUSH DE
4084  E5                 PUSH HL
4085  F5                 PUSH AF
4086  2A 7B 40           LD   HL,(407B)
4089  46                 LD   B,(HL)
408A  23                 INC  HL
```

```
408B  4E                  LD    C,(HL)
408C  E5                  PUSH  HL
408D  CD F5 08            CALL  08F5
4090  E1                  POP   HL
4091  23                  INC   HL
4092  46                  LD    B,(HL)
4093  0E 00               LD    C,00
4095  23                  INC   HL
4096  5E                  LD    E,(HL)
4097  23                  INC   HL
4098  56                  LD    D,(HL)
4099  C5         INST1    PUSH  BC
409A  D5                  PUSH  DE
409B  CD BB 02   INST2    CALL  02BB
409E  2C                  INC   L
409F  20 FA               JR    NZ INST2
40A1  2A 0E 40   INST3    LD    HL,(400E)
40A4  36 04               LD    (HL),04
40A6  06 80               LD    B,80
40A8  10 FE      INST4    DJNZ  INST4
40AA  36 00               LD    (HL),00
40AC  CD BB 02            CALL  02BB
40AF  44                  LD    B,H
40B0  4D                  LD    C,L
40B1  2C                  INC   L
40B2  28 ED               JR    Z INST3
40B4  CD BD 07            CALL  07BD
40B7  7E                  LD    A,(HL)
40B8  D1                  POP   DE
40B9  C1                  POP   BC
40BA  FE 76               CP    76
40BC  28 17               JR    Z INST6
40BE  FE 77               CP    77
40C0  28 1B               JR    Z INST7
40C2  CB 77               BIT   6,A
40C4  20 D3               JR    NZ INST1
40C6  F5                  PUSH  AF
40C7  79                  LD    A,C
40C8  B8                  CP    B
40C9  30 07               JR    NC INST5
40CB  F1                  POP   AF
40CC  D7                  RST   10
40CD  12                  LD    (DE),A
40CE  13                  INC   DE
40CF  0C                  INC   C
40D0  18 C7               JR    INST1
40D2  F1         INST5    POP   AF
40D3  18 C4               JR    INST1
40D5  3E FF      INST6    LD    A,(FF)
40D7  12                  LD    (DE),A
40D8  F1                  POP   AF
40D9  E1                  POP   HL
40DA  D1                  POP   DE
40DB  C1                  POP   BC
```

```
40DC  C9                    RET
40DD  79           INST7    LD    A,C
40DE  B7                    OR    A
40DF  28 B8                 JR    Z INST1
40E1  2A 0E 40              LD    HL,(400E)
40E4  2B                    DEC   HL
40E5  22 0E 40              LD    (400E),HL
40E8  1B                    DEC   DE
40E9  0D                    DEC   C
40AA  18 AD                 JR    INST1
```

LISTAGEM HEXADECIMAL

```
16.514      C5 D5 E5 F5 2A 7B 40 46
16.522      23 4E E5 CD F5 08 E1 23
16.530      46 0E 00 23 5E 23 56 C5
16.538      D5 CD BB 02 2C 20 FA 2A
16.546      0E 40 36 04 06 80 10 FE
16.554      36 00 CD BB 02 44 4D 2C
16.562      28 ED CD BD 07 7E D1 C1
16.570      FE 76 28 17 FE 77 28 1B
16.578      CB 77 20 D3 F5 79 B8 30
16.586      07 F1 D7 12 13 0C 18 C7
16.594      F1 18 C4 3E FF 12 F1 E1
16.602      D1 C1 C9 79 B7 28 B8 2A
16.610      0E 40 2B 22 0E 40 1B 0D
16.618      18 AD
```

DESCRICAO

Esta rotina multiplica dois numeros de 16 bits, tendo como resultado um numero de 32 bits. Como devem ser passados para esta rotina os dois operandos de 2 bytes, e devemos ter um resultado com 4 bytes, utilizamos um bloco de parametros para esta finalidade.

CONDICOES DE ENTRADA

Devemos colocar em 407Bh (16.507) e em 407Ch (16.508) o endereco do bloco de parametros, no formato padrao Z-80, primeiro o LSB seguido pelo MSB. O bloco de parametros sera formado pelo seguinte:

Byte 0:LSB do primeiro operando
Byte 1:MSB do primeiro operando
Byte 2:LSB do segundo operando
Byte 3:MSB do segundo operando
Bytes 4 a 7:Reservado para o resultado

CONDICOES DE SAIDA

O resultado estara nos bytes 4 a 7 do bloco de parametros na seguinte ordem: 1, 0, 3, 2. Os primeiros 4 bytes do bloco de parametros, 407Bh (16.507) e 407Ch (16.508) nao sao alterados.

LISTAGEM ASSEMBLY

```
4082  C5             MDPD    PUSH BC
4083  D5                     PUSH DE
4084  E5                     PUSH HL
4085  F5                     PUSH AF
4086  2A 7B 40               LD   HL,(407B)
4089  4E                     LD   C,(HL)
408A  23                     INC  HL
408B  46                     LD   B,(HL)
408C  23                     INC  HL
408D  5E                     LD   E,(HL)
408E  23                     INC  HL
408F  56                     LD   D,(HL)
4090  E5                     PUSH HL
4091  3E 10                  LD   A,10
4093  21 00 00               LD   HL,0000
4096  29             MDPD1   ADD  HL,HL
4097  EB                     EX   DE,HL
4098  ED 6A                  ADC  HL,HL
```

```
409A  EB                  EX    DE,HL
409B  30 04               JR    NC MDPD2
409D  09                  ADD   HL,BC
409E  30 01               JR    NC MDPD2
40A0  13                  INC   DE
40A1  3D          MDPD2   DEC   A
40A2  20 F3               JR    NZ MDPD1
40A4  E3                  EX    (SP),HL
40A5  23                  INC   HL
40A6  73                  LD    (HL),E
40A7  23                  INC   HL
40A8  72                  LD    (HL),D
40A9  D1                  POP   DE
40AA  23                  INC   HL
40AB  73                  LD    (HL),E
40AC  23                  INC   HL
40AD  72                  LD    (HL),D
40AE  F1                  POP   AF
40AF  E1                  POP   HL
40B0  D1                  POP   DE
40B1  C1                  POP   BC
40B2  C9                  RET
```

LISTAGEM HEXADECIMAL

```
16.514    C5 D5 E5 F5 2A 7B 40 4E
16.522    23 46 23 5E 23 56 E5 3E
16.530    10 21 00 00 29 EB ED 6A
16.538    EB 30 04 09 30 01 13 3D
16.546    20 F2 E3 23 73 23 72 D1
16.554    23 73 23 72 F1 E1 D1 C1
16.562    C9
```

DESCRICAO

Esa rotina move blocos de memoria, sendo feito uma verificacao dos enderecos da fonte e do destino para se decidir sobre a utilizacao da instrucao LDIR ou LDDR, para se evitar a sobreposicao dos blocos. Como devemos passar para esta rotina o numero de bytes a mover e os enderecos da fonte e do destino utilizamos para esta finalidade um bloco de parametros.

CONDICOES DE ENTRADA

Devemos colocar em 407Bh (16.507) e em 407Ch (16.508) o endereco do bloco de parametros, no formato padrao Z-80, primeiro o LSB seguido pelo MSB. O bloco de parametros devera ser formado pelo seguinte:

Byte 0:LSB do numero de bytes a mover
Byte 1:MSB do numero de bytes a mover
Byte 2:LSB do endereco de destino
Byte 3:MSB do endereco de destino
Byte 4:LSB do endereco da fonte
Byte 5:MSB do endereco da fonte

CONDICOES DE SAIDA

O bloco especificado e movido. O bloco de parametros, 407Bh (16.507) e 407CH (16.508) nao sao alterados.

LISTAGEM ASSEMBLY

```
4082  C5          MOBL   PUSH BC
4083  D5                 PUSH DE
4084  E5                 PUSH HL
4085  F5                 PUSH AF
4086  2A 7B 40           LD   HL,(407B)
4089  4E                 LD   C,(HL)
408A  23                 INC  HL
408B  46                 LD   B,(HL)
408C  23                 INC  HL
408D  5E                 LD   E,(HL)
408E  23                 INC  HL
408F  56                 LD   D,(HL)
4090  23                 INC  HL
4091  7E                 LD   A,(HL)
4092  23                 INC  HL
4093  66                 LD   H,(HL)
```

```
4094  6F                LD    L,A
4095  E5                PUSH  HL
4096  B7                OR    A
4097  ED 52             SBC   HL,DE
4099  CB 7C             BIT   7,H
409B  E1                POP   HL
409C  20 04             JR    NZ MOBL1
409E  ED B0             LDIR
40A0  18 08             JR    MOBL2
40A2  0B         MOBL1  DEC   BC
40A3  09                ADD   HL,BC
40A4  EB                EX    DE,HL
40A5  09                ADD   HL,BC
40A6  EB                EX    DE,HL
40A7  03                INC   BC
40A8  ED B8             LDDR
40AA  F1         MOBL2  POP   AF
40AB  E1                POP   HL
40AC  D1                POP   DE
40AD  C1                POP   BC
40AE  C9                RET
```

LISTAGEM HEXADECIMAL

```
16.514    C5 D5 E5 F5 2A 7B 40 4E
16.522    23 46 23 5E 23 56 23 7E
16.530    23 66 6F E5 B7 ED 52 CB
16.538    7C E1 20 04 ED B0 18 08
16.546    0B 09 EB 09 EB 03 ED B8
16.554    F1 E1 D1 C1 C9
```

DESCRICAO

Esta rotina multiplica dois numeros de 8 bits. Diversamente da rotina MOPO, que evita a estrutura de loop, esta rotina a utiliza, sendo inplantada com menor numeros de bytes porem aproximadamente 50% mais lenta na execucao.

CONDICOES DE,ENTRADA

Os dois operandos devem estar em 407Bh (16.507) e em 407Ch (16.508).

CONDICOES DE SAIDA

O resultado de 16 bits estara em 407Bh (16.507) e em 407Ch (16.508), no formato padrao Z-80, primeiro o LSB seguido pelo MSB

LISTAGEM ASSEMBLY

```
4082  C5          MOPL   PUSH BC
4083  D5                 PUSH DE
4084  E5                 PUSH HL
4085  2A 7B 40           LD   HL,(407B)
4088  5D                 LD   E,L
4089  16 00              LD   D,00
408B  6A                 LD   L,D
408C  06 08              LD   B,08
408E  29          MOPL1  ADD  HL,HL
408F  30 01              JR   NC MOPL2
4091  19                 ADD  HL,DE
4092  10 FA       MOPL2  DJNZ MOPL1
4094  22 7B 40           LD   (407B),HL
4097  E1                 POP  HL
4098  D1                 POP  DE
4099  C1                 POP  BC
409A  C9                 RET
```

LISTAGEM HEXADECIMAL

```
16.514     C5 D5 E5 2A 7B 40 5D 16
16.522     00 6A 06 08 29 30 01 19
16.530     10 FA 22 7B 40 E1 D1 C1
16.538     C9
```

DESCRICAO

Esta rotina multiplica dois numeros de 8 bits. Evitando uma estrutura de loop, consegue-se uma velocidade de execucao aproximadamente 50% maior, porem com um numero maior de bytes na rotina.

CONDICOES DE ENTRADA

Os dois operandos deverao estar em 407Bh (16.507) e 407Ch (16.508).

CONDICOES DE SAIDA

O resultado de 16 bits estara em 407Bh (16.507) e 407Ch (16.508) no formato padrao Z-80, primeiro o LSB seguido pelo MSB.

LISTAGEM ASSEMBLY

```
4082  C5          MOPO    PUSH BC
4083  E5                  PUSH HL
4084  2A 7B 40            LD   HL,(407B)
4087  4D                  LD   C,L
4088  06 00               LD   B,00
408A  68                  LD   L,B
408B  29                  ADD  HL,HL
408C  30 01               JR   NC MOPO1
408E  09                  ADD  HL,BC
408F  29          MOPO1   ADD  HL,HL
4090  30 01               JR   NC MOPO2
4092  09                  ADD  HL,BC
4093  29          MOPO2   ADD  HL,HL
4094  30 01               JR   NC MOPO3
4096  09                  ADD  HL,BC
4097  29          MOPO3   ADD  HL,HL
4098  30 01               JR   NC MOPO4
409A  09                  ADD  HL,BC
409B  29          MOPO4   ADD  HL,HL
409C  30 01               JR   NC MOPO5
409E  09                  ADD  HL,BC
409F  29          MOPO5   ADD  HL,HL
40A0  30 01               JR   NC MOPO6
40A2  09                  ADD  HL,BC
40A3  29          MOPO6   ADD  HL,HL
40A4  30 01               JR   NC MOPO7
40A6  09                  LD   HL,BC
```

```
40A7   29            MOPO7  ADD   HL,HL
40A8   30 01                JR    NC MOPO8
40AA   09                   ADD   HL,BC
40AB   22 7B 40      MOPO8  LD    (407B),HL
40AE   E1                   POP   HL
40AF   C1                   POP   BC
40B0   C9                   RET
```

LISTAGEM HEXADECIMAL

```
16.514     C5 E5 2A 7B 40 4D 06 00.
16.522     68 29 30 01 09 29 30 01
16.530     09 29 30 01 09 29 30 01
16.538     09 29 30 01 09 29 30 01
16.546     09 29 30 01 09 29 30 01
16.554     09 22 7B 40 E1 C1 C9
```

DESCRICAO

Esa rotina executa adicao em multipla precisao, podendo ser especificado operandos de 1 a 256 bytes, porem os dois operandos com o mesmo comprimento. Para usarmos 256 bytes devemos colocar 0 no byte 4 do bloco de parametros. Os bytes dos dois operandos deverao estar na ordem normal, nao no formato padrao Z-80, primeiro o byte mais significativo e depois os bytes menos significativos. O resultado sera colocado sobre o operando 1, e o operando 2 nao sera alterado. Como devemos passar para esa rotina o endereco dos dois operandos e o numero de bytes de ambos utilizamos para esta finalidade um bloco de parametros.

CONDICOES DE ENTRADA

Devemos colocar em 407Bh (16.507) e em 407Ch (16.508) o endereco do bloco de parametros, no formato padrao Z-80, primeiro o LSB seguido pelo MSB. O bloco de parametros devera ser formado pelo seguinte:

Byte 0:LSB do endereco do operando 1
Byte 1:MSB do endereco do operando 1
Byte 2:LSB do endereco do operando 2
Byte 3:MSB do endereco do operando 2
Byte 4:Numero de bytes dos operandos

CONDICOES DE SAIDA

O resultado da adicao em multipla precisao estara sobre o operando 1. Note que o resultado pode ser um bit maior que o numero de bytes dos dois operandos, porem esta condicao nao e mostrada. O bloco de parametros, 407Bh (16.507), 407Ch (16.508) e o segundo operando nao sao alterados.

LISTAGEM ASSEMBLY

```
4082  C5          MPAD   PUSH BC
4083  D5                 PUSH DE
4084  E5                 PUSH HL
4085  F5                 PUSH AF
4086  2A 7B 40           LD   HL,(407B)
4089  5E                 LD   E,(HL)
408A  23                 INC  HL
408B  56                 LD   D,(HL)
408C  D5                 PUSH DE
408D  23                 INC  HL
```

```
408E  5E             LD    E,(HL)
408F  23             INC   HL
4090  56             LD    D,(HL)
4091  23             INC   HL
4092  4E             LD    C,(HL)
4093  06 00          LD    B,00
4095  0B             DEC   BC
4096  E1             POP   HL
4097  09             ADD   HL,BC
4098  EB             EX    DE,HL
4099  09             ADD   HL,BC
409A  41             LD    B,C
409B  04             INC   B
409C  B7             OR    A
409D  1A      MPAD1  LD    A,(DE)
409E  8E             ADC   A,(HL)
409F  12             LD    (DE),A
40A0  2B             DEC   HL
40A1  1B             DEC   DE
40A2  10 F9          DJNZ  MPAD1
40A4  F1             POP   AF
40A5  E1             POP   HL
40A6  D1             POP   DE
40A7  C1             POP   BC
40A8  C9             RET
```

LISTAGEM HEXADECIMAL

```
16.514    C5 D5 E5 F5 2A 7B 40 5E
16.522    23 56 D5 23 5E 23 56 23
16.530    4E 06 00 0B E1 09 EB 09
16.538    41 04 B7 1A 8E 12 2B 1B
16.546    10 F9 F1 E1 D1 C1 C9
```

DESCRICAO

Esta rotina executa subtracao em multipla precisao, podendo ser especificado operandos de 1 a 256 bytes, porem os dois operandos com o mesmo comprimento. Para usarmos 256 bytes devemos colocar 0 no byte 4 do bloco de parametros. Os bytes dos dois operandos deverao estar na ordem normal, nao no formato padrao Z-80, primeiro o byte mais significativo e depois os bytes menos significativos. O resultado sera colocado sobre o operando 1, e o operando 2 nao sera alterado. Como devemos passar para esa rotina o endereco dos dois operandos e o numero de bytes de ambos utilizamos para esta finalidade um bloco de parametros.

CONDICOES DE ENTRADA

Devemos colocar em 407Bh (16.507) e em 407Ch (16.508) o endereco do bloco de parametros, no formato padrao Z-80, primeiro o LSB seguido pelo MSB. O bloco de parametros devera ser formado pelo seguinte:

Byte 0:LSB do endereco do operando 1
Byte 1:MSB do endereco do operando 1
Byte 2:LSB do endereco do operando 2
Byte 3:MSB do endereco do operando 2
Byte 4:Numero de bytes dos operandos

CONDICOES DE SAIDA

O resultado da subtracao em multipla precisao estara sobre o operando 1. Note que o resultado pode ser um bit maior que o numero de bytes dos dois operandos, porem esta condicao nao e mostrada. O bloco de parametros, 407Bh (16.507), 407Ch (16.508) e o segundo operando nao sao alterados.

LISTAGEM ASSEMBLY

```
4082  C5         MPSB   PUSH BC
4083  D5                PUSH DE
4084  E5                PUSH HL
4085  F5                PUSH AF
4086  2A 7B 40          LD   HL,(407B)
4089  5E                LD   E,(HL)
408A  23                INC  HL
408B  56                LD   D,(HL)
408C  D5                PUSH DE
```

```
408D   23              INC   HL
408E   5E              LD    E,(HL)
408F   23              INC   HL
4090   56              LD    D,(HL)
4091   23              INC   HL
4092   4E              LD    C,(HL)
4093   06 00           LD    B,00
4095   0B              DEC   BC
4096   E1              POP   HL
4097   09              ADD   HL,BC
4098   EB              EX    DE,HL
4099   09              ADD   HL,BC
409A   41              LD    B,C
409B   04              INC   B
409C   B7              OR    A
409D   1A        MPSB1 LD    A,(DE)
409E   9E              SBC   A,(HL)
409F   12              LD    (DE),A
40A0   2B              DEC   HL
40A1   1B              DEC   DE
40A2   10 F9           DJNZ  MPSB1
40A4   F1              POP   AF
40A5   E1              POP   HL
40A6   D1              POP   DE
40A7   C1              POP   BC
40A8   C9              RET
```

LISTAGEM HEXADECIMAL

```
16.514    C5 D5 E5 F5 2A 7B 40 5E
16.522    23 56 D5 23 5E 23 56 23
16.530    4E 06 00 0B E1 09 EB 09
16.538    41 04 B7 1A 9E 12 2B 1B
16.546    10 F9 F1 E1 D1 C1 C9
```

DESCRICAO

Esta rotina efetua multiplos deslocamentos a esquerda em um numero de 2 bytes. Deslocamento a esquerda consiste em deslocar bit a bit o numero para a esquerda, sendo o bit mais a esquerda colocado no CARRY FLAG e o bit mais a direita preenchido com 0. Se especificarmos 0 no numero de deslocamentos, sera efetuado 256 deslocamentos e o resultado sera 0; e se o numero de deslocamentos estiver entre 16 e 255 o resultado tambem sera 0.

CONDICOES DE ENTRADA

Devemos colocar em 407Bh (16.507) e em 407Ch (16.508) o numero de 2 bytes a ser deslocado, no formato padrao Z-80, primeiro o LSB seguido do MSB. O numero de deslocamentos devera ser colocado em 4021h (16.417).

CONDICOES DE SAIDA

407Bh (16.507) e 407Ch (16.508) contem o numero deslocado. 4021h (16.417) nao sera alterado.

LISTAGEM ASSEMBLY

```
4082  C5           MSLF    PUSH BC
4083  E5                   PUSH HL
4084  F5                   PUSH AF
4085  2A 7B 40             LD   HL,(407B)
4088  3A 21 40             LD   A,(4021)
408B  47                   LD   B,A
408C  29           MSLF1   ADD  HL,HL
408D  10 FD                DJNZ MSLF1
408F  22 7B 40             LD   (407B),HL
4092  F1                   POP  AF
4093  E1                   POP  HL
4094  C1                   POP  BC
4095  C9                   RET
```

LISTAGEM HEXADECIMAL

```
16.514    C5 E5 F5 2A 7B 40 3A 21
16.522    40 47 29 10 FD 22 7B 40
16.530    F1 E1 C1 C9
```

DESCRICAO

Esta rotina efetua multiplos deslocamentos a direita em um numero de 2 bytes. Deslocamento a direita consiste em deslocar bit a bit o numero para a direita, sendo o bit mais a direita colocado no CARRY FLAG e o bit mais a esquerda preenchido com 0. Se especificarmos 0 no numero de deslocamentos, sera efetuado 256 deslocamentos e o resultado sera 0; e se o numero de deslocamentos estiver entre 16 e 255 o resultado tambem sera 0.

CONDICOES DE ENTRADA

Devemos colocar em 407Bh (16.507) e em 407Ch (16.508) o numero de 2 bytes a ser deslocado, no formato padrao Z-80, primeiro o LSB seguido do MSB. O numero de deslocamentos devera ser colocado em 4021h (16.417).

CONDICOES DE SAIDA

407Bh (16.507) e 407Ch (16.508) contem o numero deslocado. 4021h (16.417) nao sera alterado.

LISTAGEM ASSEMBLY

```
4082  C5              MSRG    PUSH BC
4083  E5                      PUSH HL
4084  F5                      PUSH AF
4085  2A 7B 40                LD   HL,(407B)
4088  3A 21 40                LD   A,(4021)
408B  47                      LD   B,A
408C  CB 3C           MSRG1   SRL  H
408E  CB 1D                   RR   L
4090  10 FA                   DJNZ MSRG1
4092  22 7B 40                LD   (407B),HL
4095  F1                      POP  AF
4096  E1                      POP  HL
4097  C1                      POP  BC
4098  C9                      RET
```

LISTAGEM HEXADECIMAL

```
16.514     C5 E5 F5 2A 7B 40 3A 21
16.522     40 47 CB 3C CB 1D 10 FA
16.530     22 7B 40 F1 E1 C1 C9
```

DESCRICAO

Esta rotina preenche um bloco de memoria com um caractere especificado. Como devem ser passados para esta rotina o caractere a ser colocado, o endereco inicial e o numero de bytes do bloco de memoria, 5 bytes no total, utilizamos um bloco de parametros para esta finalidade.

CODICOES DE ENTRADA

407Bh (16.507) e 407Ch (16.508) devem apontar para um bloco de parametros formado pelo seguinte:

Byte 0:Caractere a ser colocado
Byte 1:LSB do endereco do bloco de memoria
Byte 2:MSB do endereco do bloco de memoria
Byte 3:LSB do numero de bytes do bloco de memoria
Byte 4:MSB do numero de bytes do bloco de memoria

CONDICOES DE SAIDA

O bloco de memoria sera preenchido com o caractere desejado. 407Bh (16.507), 407Ch (16.508) e o bloco de parametros nao sao alterados.

LISTAGEM ASSEMBLY

```
4082  C5          PRME    PUSH BC
4083  D5                  PUSH DE
4084  E5                  PUSH HL
4085  F5                  PUSH AF
4086  2A 7B 40            LD   HL,(407B)
4089  7E                  LD   A,(HL)
408A  23                  INC  HL
408B  5E                  LD   E,(HL)
408C  23                  INC  HL
408D  56                  LD   D,(HL)
408E  23                  INC  HL
408F  4E                  LD   C,(HL)
4090  23                  INC  HL
4091  46                  LD   B,(HL)
4092  EB                  EX   DE,HL
4093  77          PRME1   LD   (HL),A
4094  23                  INC  HL
4095  0B                  DEC  BC
4096  57                  LD   D,A
4097  78                  LD   A,B
4098  B1                  OR   C
```

```
4099  7A          LD   A,D
409A  20 F7       JR   NZ PRME1
409C  F1          POP  AF
409D  E1          POP  HL
409E  D1          POP  DE
409F  C1          POP  BC
40A0  C9          RET
```

LISTAGEM HEXADECIMAL

```
16.514    C5 D5 E5 F5 2A 7B 40 7E
16.522    23 5E 23 56 23 4E 23 46
16.530    EB 77 23 0B 57 78 B1 7A
16.538    20 F7 F1 E1 D1 C1 C9
```

DESCRICAO

Esta rotina preenche linhas do video, devendo ser especificado a linha inicial, a linha final e o caractere a ser colocado no video. Nao e verificado se os numeros de linha fornecidos estao dentro da faixa permitida, que e de 00h a 17h (0 a 23), se o numero final e maior que o inicial, e se o caractere especificado pode ser colocado no video. Se estas condicoes nao forem seguidas o computador ficara fora de controle, perdendo-se todos os programas da memoria.

CONDICOES DE ENTRADA

O numero da linha inicial devera ser colocado em 407Bh (16.507), o numero da linha final em 407Ch (16.508), e o caractere desejado em 4021h (16.417).

CONDICOES DE SAIDA

As linhas do video especificadas serao preenchidas com o caractere indicado. 407Bh (16.507) , 407Ch (16.508) e 4021h (16.417) nao sao alterados.

LISTAGEM ASSEMBLY

```
4082  C5          PSLN   PUSH BC
4083  D5                 PUSH DE
4084  E5                 PUSH HL
4085  F5                 PUSH AF
4086  2A 7B 40           LD   HL,(407B)
4089  E5                 PUSH HL
408A  7C                 LD   A,H
408B  95                 SUB  L
408C  3C                 INC  A
408D  6F                 LD   L,A
408E  26 00              LD   H,00
4090  E5                 PUSH HL
4091  29                 ADD  HL,HL
4092  29                 ADD  HL,HL
4093  29                 ADD  HL,HL
4094  29                 ADD  HL,HL
4095  29                 ADD  HL,HL
4096  C1                 POP  BC
4097  09                 ADD  HL,BC
4098  EB                 EX   HL,BC
4099  E1                 POP  HL
```

```
409A  26 00              LD   H,00
409C  E5                 PUSH HL
409D  29                 ADD  HL,HL
409E  29                 ADD  HL,HL
409F  29                 ADD  HL,HL
40A0  29                 ADD  HL,HL
40A1  29                 ADD  HL,HL
40A2  C1                 POP  BC
40A3  09                 ADD  HL,BC
40A4  44                 LD   B,H
40A5  4D                 LD   C,L
40A6  2A 0C 40           LD   HL,(400C)
40A9  09                 ADD  HL,BC
40AA  3A 21 40           LD   A,(4021)
40AD  4F                 LD   C,A
40AE  23          PSLN1  INC  HL
40AF  7E                 LD   A,(HL)
40B0  FE 76              CP   76
40B2  28 01              JR   Z PSLN2
40B4  71                 LD   (HL),C
40B5  1B          PSLN2  DEC  DE
40B6  7A                 LD   A,D
40B7  B3                 OR   E
40B8  20 F4              JR   NZ PSLN1
40BA  F1                 POP  AF
40BB  E1                 POP  HL
40BC  D1                 POP  DE
40BD  C1                 POP  BC
40BE  C9                 RET
```

LISTAGEM HEXADECIMAL

```
16.514    C5 D5 E5 F5 2A 7B 40 E5
16.522    7C 95 3C 6F 26 00 E5 29
16.530    29 29 29 29 C1 09 EB E1
16.538    26 00 E5 29 29 29 29 29
16.546    C1 09 44 4D 2A 0C 40 09
16.554    3A 21 40 4F 23 7E FE 76
16.562    28 01 71 1B 7A B3 20 F4
16.570    F1 E1 D1 C1 C9
```

DESCRICAO

Esta rotina imprime no video uma string, devendo ser especificado a linha e coluna para se iniciar a impressao. A string pode ter qualquer numero de bytes, porem deve ser terminada pelo byte FFh (255). Como devemos passar para esta rotina os numeros da linha e coluna para impressao e o endereco da string, utilizamos para esta finalidade um bloco de parametros.

CONDICOES DE ENTRADA

Devemos colocar em 407Bh (16.507) e em 407Ch (16.508) o endereco do bloco de parametros, no formato padrao Z-80, primeiro o LSB seguido pelo MSB. O bloco de parametros deve ser formado pelo seguinte:

Byte 0:Numero da linha para impressao
Byte 1:Numero da coluna para impressao
Byte 2:LSB do endereco da string
Byte 3:MSB do endereco da string

CONDICOES DE SAIDA

A string e impressa no video. O bloco de parametros, 407Bh (16.507) e 407Ch (16.508) nao sao alterados.

LISTAGEM ASSEMBLY

```
4082  C5               PTST    PUSH BC
4083  D5                       PUSH DE
4084  E5                       PUSH HL
4085  F5                       PUSH AF
4086  2A 7B 40                 LD   HL,(407B)
4089  46                       LD   B,(HL)
408A  23                       INC  HL
408B  4E                       LD   C,(HL)
408C  E5                       PUSH HL
408D  CD F5 08                 CALL 08F5
4090  E1                       POP  HL
4091  23                       INC  HL
4092  5E                       LD   E,(HL)
4093  23                       INC  HL
4094  56                       LD   D,(HL)
4095  1A               PTST1   LD   A,(DE)
4096  FE FF                    CP   FF
4098  28 04                    JR   Z PTST2
```

```
409A  D7          RST  10
409B  13          INC  DE
409C  18 F7       JR   PTST1
409E  F1          POP  AF
409F  E1          POP  HL
40A0  D1          POP  DE
40A1  C1          POP  BC
40A2  C9          RET
```

LISTAGEM HEXADECIMAL

```
16.514     C5 D5 E5 F5 2A 7B 40 46
16.522     23 4E E5 CD F5 08 E1 23
16.530     5E 23 56 1A FE FF 28 04
16.538     D7 13 18 F7 F1 E1 D1 C1
16.546     C9
```

DESCRICAO

Esta rotina roda as 24 linhas do video uma posicao para baixo, sendo o conteudo da linha 23 copiado na linha 0.

CONDICOES DE ENTRADA

Nenhuma.

CONDICOES DE SAIDA

O video e rodado uma posicao para baixo. O buffer da impressora 403Ch a 405Bh (16.444 a 16.475) e utilizado como armazenamento temporario e portanto e alterado.

LISTAGEM ASSEMBLY

```
4082  C5           RVPB  PUSH BC
4083  D5                 PUSH DE
4084  E5                 PUSH HL
4085  2A 0C 40           LD   HL,(400C)
4088  01 18 03           LD   BC,0318
408B  09                 ADD  HL,BC
408C  E5                 PUSH HL
408D  2B                 DEC  HL
408E  11 5B 40           LD   DE,405B
4091  01 20 00           LD   BC,0020
4094  ED B8              LDDR
4096  D1                 POP  DE
4097  01 F7 02           LD   BC,02F7
409A  ED B8              LDDR
409C  1B                 DEC  DE
409D  21 5B 40           LD   HL,405B
40A0  01 20 00           LD   BC,0020
40A3  ED B8              LDDR
40A5  E1                 POP  HL
40A6  D1                 POP  DE
40A7  C1                 POP  BC
40A8  C9                 RET
```

LISTAGEM HEXADECIMAL

```
16.514    C5 D5 E5 2A 0C 40 01 18
16.522    03 09 E5 2B 11 5B 40 01
16.530    20 00 ED B8 D1 01 F7 02
16.538    ED B8 1B 21 5B 40 01 20
16.546    00 ED B8 E1 D1 C1 C9
```

DESCRICAO

Esta rotina roda as 24 linhas do video uma posicao para cima, sendo que o conteudo da linha 0 e copiado na linha 23.

CONDICOES DE ENTRADA

Nenhuma.

CONDICOES DE SAIDA

O video e rodado uma posicao para cima. O buffer da impressora 403Ch a 405Bh (16.444 a 16475) e utilizado como armazenamento temporario e portanto e alterado.

LISTAGEM ASSEMBLY

```
4082  C5          RVPC  PUSH BC
4083  D5                PUSH DE
4084  E5                PUSH HL
4085  2A 0C 40          LD   HL,(400C)
4088  E5                PUSH HL
4089  23                INC  HL
408A  11 3C 40          LD   DE,403C
408D  01 20 00          LD   BC,0020
4090  ED B0             LDIR
4092  D1                POP  DE
4093  01 F7 02          LD   BC,02F7
4096  ED B0             LDIR
4098  13                INC  DE
4099  21 3C 40          LD   HL,403C
409C  01 20 00          LD   BC,0020
409F  ED B0             LDIR
40A1  E1                POP  HL
40A2  D1                POP  DE
40A3  C1                POP  BC
40A4  C9                RET
```

LISTAGEM HEXADECIMAL

```
16.514    C5 D5 E5 2A 0C 40 E5 23
16.522    11 3C 40 01 20 00 ED B0
16.530    D1 01 F7 02 ED B0 13 21
16.538    3C 40 01 20 00 ED B0 E1
16.546    D1 C1 C9
```

* * * R V P D * * *

DESCRICAO

Esta rotina roda as 24 linhas do video uma posicao para a direita, sendo o conteudo da margem direita copiado na margem esquerda.

CONDICOES DE ENTRADA

Nenhuma.

CONDICOES DE SAIDA

O video e rodado uma posicao para a direita

LISTAGEM ASSEMBLY

```
4082  C5          RVPD   PUSH BC
4083  D5                 PUSH DE
4084  E5                 PUSH HL
4085  F5                 PUSH AF
4086  2A 0C 40           LD   HL,(400C)
4089  01 17 03           LD   BC,0317
408C  09                 ADD  HL,BC
408D  54                 LD   D,H
408E  5D                 LD   E,L
408F  2B                 DEC  HL
4090  06 18              LD   B,18
4092  1A          RVPD1  LD   A,(DE)
4093  F5                 PUSH AF
4094  7E          RVPD2  LD   A,(HL)
4095  FE 76              CP   76
4097  28 05              JR   Z RVPD3
4099  12                 LD   (DE),A
409A  2B                 DEC  HL
409B  1B                 DEC  DE
409C  18 F6              JR   RVPD2
409E  F1          RVPD3  POP  AF
409F  12                 LD   (DE),A
40A0  2B                 DEC  HL
40A1  2B                 DEC  HL
40A2  1B                 DEC  DE
40A3  1B                 DEC  DE
40A4  10 EC              DJNZ RVPD1
40A6  F1                 POP  AF
40A7  E1                 POP  HL
40A8  D1                 POP  DE
40A9  C1                 POP  BC
```

```
40AA   C9                          RET
```

LISTAGEM HEXADECIMAL

```
16.514     C5 D5 E5 F5 2A 0C 40 01
16.522     17 03 09 54 5D 2B 06 18
16.530     1A F5 7E FE 76 28 05 12
16.538     2B 1B 18 F6 F1 12 2B 2B
16.546     1B 1B 10 EC F1 E1 D1 C1
16.554     C9
```

DESCRICAO

Esta rotina roda as 24 linhas do video uma posicao para a esquerda, sendo que o conteudo da margem esquerda e copiado na margem direita.

CONDICOES DE ENTRADA

Nenhuma.

CONDICOES DE SAIDA

O video sera rodado uma posicao para a esquerda.

LISTAGEM ASSEMBLY

```
4082  C5            RVPE   PUSH BC
4083  D5                   PUSH DE
4084  E5                   PUSH HL
4085  F5                   PUSH AF
4086  2A 0C 40             LD   HL,(400C)
4089  23                   INC  HL
408A  54                   LD   D,H
408B  5D                   LD   E,L
408C  23                   INC  HL
408D  06 18                LD   B,18
408F  1A            RVPE1  LD   A,(DE)
4090  F5                   PUSH AF
4091  7E            RVPE2  LD   A,(HL)
4092  FE 76                CP   76
4094  28 05                JR   Z RVPE3
4096  12                   LD   (DE),A
4097  23                   INC  HL
4098  13                   INC  DE
4099  18 F6                JR   RVPD2
409B  F1            RVPE3  POP  AF
409C  12                   LD   (DE),A
409D  23                   INC  HL
409E  23                   INC  HL
409F  13                   INC  DE
40A0  13                   INC  DE
40A1  10 EC                DJNZ RVPE1
40A3  F1                   POP  AF
40A4  E1                   POP  HL
40A5  D1                   POP  DE
40A6  C1                   POP  BC
40A7  C9                   RET
```

LISTAGEM HEXADECIMAL

```
16.514      C5 D5 E5 F5 2A 0C 40 23
16.522      54 5D 23 06 18 1A F5 7E
16.530      FE 76 28 05 12 23 13 18
16.538      F6 F1 12 23 23 13 13 10
16.546      EC F1 E1 D1 C1 C9
```

DESCRICAO

Esta rotina desloca o video para baixo e para a esquerda simultaneamente. O numero da linha a partir da qual sera feito o scroll devera ser colocado na posicao 407Bh (16.507), ou seja, se Voce colocar nesta posicao o numero 15, o scroll sera executado da linha 15 ate a linha 23,sendo o conteudo da linha 23 perdido e a linha 15 preenchida com espacos. Este numero de linha devera estar entre 00h e 16h (0 e 22) ou a rotina nao funcionara. O conteudo da margem esquerda sera perdido e a margem direita sera preenchida com espacos.

CONDICOES DE ENTRADA

Colocar em 407Bh (16.507) o numero da linha a partir da qual sera executado o scroll, entre 00h e 16h (0 e 22).

CONDICOES DE SAIDA

O video sera deslocado uma linha para baixo, a partir da linha especificada, e uma posicao para a esquerda. 407Bh (16.507) nao sera alterado.

LISTAGEM ASSEMBLY

```
4082  C5          SCDL    PUSH BC
4083  D5                  PUSH DE
4084  E5                  PUSH HL
4085  F5                  PUSH AF
4086  21 7B 40            LD   HL,407B
4089  3E 17               LD   A,17
408B  96                  SUB  (HL)
408C  2A 0C 40            LD   HL,(400C)
408F  11 17 03            LD   DE,0317
4092  19                  ADD  HL,DE
4093  36 00               LD   (HL),00
4095  11 E1 FF    SCDL1   LD   DE,FFE1
4098  19                  ADD  HL,DE
4099  54                  LD   D,H
409A  5D                  LD   E,L
409B  01 E0 FF            LD   BC,FFE0
409E  09                  ADD  HL,BC
409F  01 1F 00            LD   BC,001F
40A2  ED B0               LDIR
40A4  2B                  DEC  HL
```

```
40A5   36 00              LD    (HL),00
40A7   3D                 DEC   A
40A8   20 EB              JR    NZ SCDL1
40AA   06 20              LD    B,20
40AC   36 00       SCUL2  LD    (HL),00
40AE   2B                 DEC   HL
40AF   10 FB              DJNZ  SCDL2
40B1   F1                 POP   AF
40B2   E1                 POP   HL
40B3   D1                 POP   DE
40B4   C1                 POP   BC
40B5   C9                 RET
```

LISTAGEM HEXADECIMAL

```
16.514    C5 D5 E5 F5 21 7B 40 3E
16.522    17 96 2A 0C 40 11 17 03
16.530    19 36 00 11 E1 FF 19 54
16.538    5D 01 E0 FF 09 01 1F 00
16.546    ED B0 2B 36 00 3D 20 EB
16.554    06 20 36 00 2B 10 FB F1
16.562    E1 D1 C1 C9
```

DESCRICAO

Esta rotina desloca o video para baixo e para a direita simultaneamente. O numero da linha a partir da qual sera feito o scroll devera ser colocado na posicao 407Bh (16.507), ou seja, se Voce colocar nesta posicao o numero 15, o scroll sera executado da linha 15 ate a linha 23,sendo o conteudo da linha 23 perdido e a linha 15 preenchida com espacos. Este numero de linha devera estar entre 00h e 16h (1 e 22) ou a rotina nao funcionara. O conteudo da margem direita sera perdido e a margem esquerda sera preenchida com espacos.

CONDICOES DE ENTRADA

Colocar em 407Bh (16.507) o numero da linha a partir da qual sera executado o scroll, entre 00h e 16h (0 e 22).

CONDICOES DE SAIDA

O video sera deslocado uma linha para baixo, a partir da linha especificada, e uma posicao para a direita 407Bh (16.507) nao sera alterado.

LISTAGEM ASSEMBLY

```
4082  C5          SCDR   PUSH BC
4083  D5                 PUSH DE
4084  E5                 PUSH HL
4085  F5                 PUSH AF
4086  21 7B 40           LD   HL,407B
4089  3E 17              LD   A,17
408B  96                 SUB  (HL)
408C  2A 0C 40           LD   HL,(400C)
408F  11 F8 02           LD   DE,02F8
4092  19                 ADD  HL,DE
4093  36 00              LD   (HL),00
4095  11 1F 00    SCDR1  LD   DE,001F
4098  19                 ADD  HL,DE
4099  54                 LD   D,H
409A  5D                 LD   E,L
409B  01 DE FF           LD   BC,FFDE
409E  09                 ADD  HL,BC
409F  01 1F 00           LD   BC,001F
40A2  ED B8              LDDR
40A4  23                 INC  HL
```

```
40A5   36 00            LD    (HL),00
40A7   3D               DEC   A
40A8   20 EB            JR    NZ SCDR1
40AA   06 20            LD    B,20
40AC   36 00      SCDR2 LD    (HL),00
40AE   23               INC   HL
40AF   10 FB            DJNZ  SCDR2
40B1   F1               POP   AF
40B2   E1               POP   HL
40B3   D1               POP   DE
40B4   C1               POP   BC
40B5   C9               RET
```

LISTAGEM HEXADECIMAL

```
16.514     C5 D5 E5 F5 21 7B 40 3E
16.522     17 96 2A 0C 40 11 F8 02
16.530     19 36 00 11 1F 00 19 54
16.538     5D 01 DE FF 09 01 1F 00
16.546     ED B8 23 36 00 3D 20 EB
16.554     06 20 36 00 23 10 FB F1
16.562     E1 D1 C1 C9
```

DESCRICAO

Esta rotina desloca o video para baixo uma linha, devendo ser especificado a partir de qual linha devera ser feito o scroll. Se especificarmos, por exemplo, a linha 15 para inicio do scroll, o video sera deslocado uma linha para baixo, sendo que o conteudo da linha 23 sera perdido e a linha 15 preenchida com espacos.

CONDICOES DE ENTRADA

Devemos colocar em 407Bh (16.507) o numero da linha a partir da qual sera feito o scroll, entre 00h e 16h (0 e 22).

CONDICOES DE SAIDA

O video e deslocado uma linha para baixo, a partir da linha especificada. O cursor nao e posicionado para a esta linha. 407Bh (16.507) nao e alterado.

LISTAGEM ASSEMBLY

```
4082  C5          SCDW   PUSH BC
4083  D5                 PUSH DE
4084  E5                 PUSH HL
4085  21 7B 40           LD   HL,407B
4088  3E 17              LD   A,17
408A  96                 SUB  (HL)
408B  6F                 LD   L,A
408C  26 00              LD   H,00
408E  44                 LD   B,H
408F  4D                 LD   C,L
4090  29                 ADD  HL,HL
4091  29                 ADD  HL,HL
4092  29                 ADD  HL,HL
4093  29                 ADD  HL,HL
4094  29                 ADD  HL,HL
4095  09                 ADD  HL,BC
4096  E5                 PUSH HL
4097  2A 0C 40           LD   HL,(400C)
409A  01 18 03           LD   BC,0318
409D  09                 ADD  HL,BC
409E  54                 LD   D,H
409F  5D                 LD   E,L
40A0  01 DF FF           LD   BC,FFDF
40A3  09                 ADD  HL,BC
```

```
40A4   C1              POP   BC
40A5   ED B8           LDDR
40A7   06 20           LD    B,20
40A9   23      SCDW1   INC   HL
40AA   36 00           LD    (HL),00
40AC   10 FB           DJNZ  SCDW1
40AE   E1              POP   HL
40AF   D1              POP   DE
40B0   C1              POP   BC
40B1   C9              RET
```

LISTAGEM HEXADECIMAL

```
16.514   C5 D5 E5 21 7B 40 3E 17
16.522   96 6F 26 00 44 4D 29 29
16.530   29 29 29 09 E5 2A 0C 40
16.538   01 18 03 09 54 5D 01 DF
16.546   FF 09 C1 ED B8 06 20 23
16.554   36 00 10 FB E1 D1 C1 C9
```

DESCRICAO

Esta rotina desloca as 24 linhas do video uma posicao para a esquerda, sendo o conteudo da margem esquerda perdido, e a margem direita preenchida com espacos.

CONDICOES DE ENTRADA

Nenhuma.

CONDICOES DE SAIDA

O video sera deslocado uma posicao para a esquerda.

LISTAGEM ASSEMBLY

```
4082  C5           SCLT   PUSH BC
4083  D5                  PUSH DE
4084  E5                  PUSH HL
4085  F5                  PUSH AF
4086  2A 0C 40            LD   HL,(400C)
4089  23                  INC  HL
408A  54                  LD   D,H
408B  5D                  LD   E,L
408C  23                  INC  HL
408D  06 18               LD   B,18
408F  7E           SCLT1  LD   A,(HL)
4090  FE 76               CP   76
4092  28 05               JR   Z SCLT2
4094  12                  LD   (DE),A
4095  23                  INC  HL
4096  13                  INC  DE
4097  18 F6               JR   SCLT1
4099  AF           SCLT2  XOR  A
409A  12                  LD   (DE),A
409B  23                  INC  HL
409C  23                  INC  HL
409D  13                  INC  DE
409E  13                  INC  DE
409F  10 EE               DJNZ SCLT1
40A1  F1                  POP  AF
40A2  E1                  POP  HL
40A3  D1                  POP  DE
40A4  C1                  POP  BC
40A5  C9                  RET
```

LISTAGEM HEXADECIMAL

```
16.514      C5 D5 E5 F5 2A 0C 40 23
16.522      54 5D 23 06 18 7E FE 76
16.530      28 05 12 23 13 18 F6 AF
16.538      12 23 23 13 13 10 EE F1
16.546      E1 D1 C1 C9
```

DESCRICAO

Esta rotina desloca as 24 linhas do video uma posicao para a direita, sendo o conteudo da margem direita perdido, e a margem esquerda preenchida com espacos.

CONDICOES DE ENTRADA

Nenhuma.

CONDICOES DE SAIDA

O video sera deslocado uma posicao para a direita.

LISTAGEM ASSEMBLY

```
4082  C5          SCRG   PUSH BC
4083  D5                 PUSH DE
4084  E5                 PUSH HL
4085  F5                 PUSH AF
4086  2A 0C 40           LD   HL,(400C)
4089  01 17 03           LD   BC,0317
408C  09                 ADD  HL,BC
408D  54                 LD   D,H
408E  5D                 LD   E,L
408F  2B                 DEC  HL
4090  06 18              LD   B,18
4092  7E          SCRG1  LD   A,(HL)
4093  FE 76              CP   76
4095  28 05              JR   Z SCRG2
4097  12                 LD   (DE),A
4098  2B                 DEC  HL
4099  1B                 DEC  DE
409A  18 F6              JR   SCRG1
409C  AF                 XOR  A
409D  12                 LD   (DE),A
409E  2B                 DEC  HL
409F  2B                 DEC  HL
40A0  1B                 DEC  DE
40A1  1B                 DEC  DE
40A2  10 EE              DJNZ SCRG1
40A4  F1                 POP  AF
40A5  E1                 POP  HL
40A6  D1                 POP  DE
40A7  C1                 POP  BC
40A8  C9                 RET
```

LISTAGEM HEXADECIMAL

16.514	C5 D5 E5 F5 2A 0C 40 01
16.522	17 03 09 54 5D 2B 06 1B
16.530	7E FE 76 28 05 12 2B 1B
16.538	18 F6 AF 12 2B 2B 1B 1B
16.546	10 EE F1 E1 D1 C1 C9

DESCRICAO

Esta rotina desloca o video para cima e para a esquerda simultaneamente. O numero da linha a partir da qual sera feito o scroll devera ser colocado na posicao 407Bh (16.507), ou seja, se Voce colocar nesta posicao o numero 15, o scroll sera executado da linha 15 ate a linha 0, sendo o conteudo da linha 0 perdido e a linha 15 preenchida com espacos. Este numero de linha devera estar entre 01h e 17h (1 e 23) ou a rotina nao funcionara. O conteudo da margem esquerda sera perdido e a margem direita sera preenchida com espacos.

CONDICOES DE ENTRADA

Colocar em 407Bh (16.507) o numero da linha a partir da qual sera executado o scroll, entre 01h e 17h (1 e 23.

CONDICOES DE SAIDA

O video sera deslocado uma linha para cima, a partir da linha especificada, e uma posicao para a esquerda. 407Bh (16.507) nao sera alterado.

LISTAGEM ASSEMBLY

```
4082  C5          SCUL   PUSH BC
4083  D5                 PUSH DE
4084  E5                 PUSH HL
4085  F5                 PUSH AF
4086  3A 7B 40           LD   A,(407B)
4089  2A 0C 40           LD   HL,(400C)
408C  E5                 PUSH HL
408D  11 20 00           LD   DE,0020
4090  19                 ADD  HL,DE
4091  36 00              LD   (HL),00
4093  E1                 POP  HL
4094  11 01 00           LD   DE,0001
4097  19          SCUL1  ADD  HL,DE
4098  54                 LD   D,H
4099  5D                 LD   E,L
409A  01 22 00           LD   BC,0022
409D  09                 ADD  HL,BC
409E  01 1F 00           LD   BC,001F
40A1  ED B0              LDIR
40A3  2B                 DEC  HL
```

```
40A4   36 00              LD    (HL),00
40A6   11 E1 FF           LD    DE,FFE1
40A9   3D                 DEC   A
40AA   20 EB              JR    NZ SCUL1
40AC   06 20              LD    B,20
40AE   36 00       SCUL2  LD    (HL),00
40B0   2B                 DEC   HL
40B1   10 FB              DJNZ  SCUL2
40B3   F1                 POP   AF
40B4   E1                 POP   HL
40B5   D1                 POP   DE
40B6   C1                 POP   BC
40B7   C9                 RET
```

LISTAGEM HEXADECIMAL

```
16.514     C5 D5 E5 F5 3A 7B 40 2A
16.522     0C 40 E5 11 20 00 19 36
16.530     00 E1 11 01 00 19 54 5D
16.538     01 22 00 09 01 1F 00 ED
16.546     B0 2B 36 00 11 E1 FF 3D
16.554     20 EB 06 20 36 00 2B 10
16.562     FB F1 E1 D1 C1 C9
```

DESCRICAO

Esta rotina desloca o video para cima uma linha, devendo ser especificado a partir de qual linha devera ser feito o scroll. Se especificarmos, por exemplo, a linha 15 para inicio do scroll, o video sera deslocado uma linha para cima, sendo que o conteudo da linha 0 sera perdido e a linha 15 preenchida com espacos.

CONDICOES DE ENTRADA

Devemos colocar em 407Bh (16.507) o numero da linha a partir da qual sera feito o scroll, entre 01h e 17h (1 e 23).

CONDICOES DE SAIDA

O video e deslocado uma linha para cima, a partir da linha especificada. O cursor nao e posicionado para a esta linha. 407Bh (16.507) nao e alterado.

LISTAGEM ASSEMBLY

```
4082  C5            SCUP    PUSH BC
4083  D5                    PUSH DE
4084  E5                    PUSH HL
4085  2A 7B 40              LD   HL,(407B)
4088  26 00                 LD   H,00
408A  44                    LD   B,H
408B  4D                    LD   C,L
408C  29                    ADD  HL,HL
408D  29                    ADD  HL,HL
408E  29                    ADD  HL,HL
408F  29                    ADD  HL,HL
4090  29                    ADD  HL,HL
4091  09                    ADD  HL,BC
4092  E5                    PUSH HL
4093  2A 0C 40              LD   HL,(400C)
4096  54                    LD   D,H
4097  5D                    LD   E,L
4098  01 21 00              LD   BC,0021
409B  09                    ADD  HL,BC
409C  C1                    POP  BC
409D  ED B0                 LDIR
409F  06 20                 LD   B,20
40A1  2B            SCUP1   DEC  HL
40A2  36 00                 LD   (HL),00
```

```
40A4   10 FB            DJNZ SCUP1
40A6   E1               POP  HL
40A7   D1               POP  DE
40A8   C1               POP  BC
40A9   C9               RET
```

LISTAGEM HEXADECIMAL

```
16.514      C5 D5 E5 2A 7B 40 26 00
16.522      44 4D 29 29 29 29 29 09
16.530      E5 2A 0C 40 54 5D 01 21
16.538      00 09 C1 ED B0 06 20 2B
16.546      36 00 10 FB E1 D1 C1 C9
```

DESCRICAO

Esta rotina desloca o video para cima e para a direita simultaneamente. O numero da linha a partir da qual sera feito o scroll devera ser colocado na posicao 407Bh (16.507), ou seja, se Voce colocar nesta posicao o numero 15, o scroll sera executado da linha 15 ate a linha 0, sendo o conteudo da linha 0 perdido e a linha 15 preenchida com espacos. Este numero de linha devera estar entre 01h e 17h (1 e 23) ou a rotina nao funcionara. O conteudo da margem direita sera perdido e a margem esquerda sera preenchida com espacos.

CONDICOES DE ENTRADA

Colocar em 407Bh (16.507) o numero da linha a partir da qual sera executado o scroll, entre 01h e 17h (1 e 23).

CONDICOES DE SAIDA

O video sera deslocado uma linha para cima, a partir da linha especificada, e uma posicao para a direita. 407Bh (16.507) nao sera alterado.

LISTAGEM ASSEMBLY

```
4082  C5          SCUR   PUSH BC
4083  D5                 PUSH DE
4084  E5                 PUSH HL
4085  F5                 PUSH AF
4086  3A 7B 40           LD   A,(407B)
4089  2A 0C 40           LD   HL,(400C)
408C  23                 INC  HL
408D  36 00              LD   (HL),00
408F  11 1F 00    SCUR1  LD   DE,001F
4092  19                 ADD  HL,DE
4093  54                 LD   D,H
4094  5D                 LD   E,L
4095  01 20 00           LD   BC,0020
4098  09                 ADD  HL,BC
4099  0B                 DEC  BC
409A  ED B8              LDDR
409C  23                 INC  HL
409D  36 00              LD   (HL),00
409F  3D                 DEC  A
40A0  20 ED              JR   NZ SCUR1
```

```
40A2   06 20              LD   B,20
40A4   36 00       SCUR2  LD   (HL),00
40A6   23                 INC  HL
40A7   10 FB              DJNZ SCUR2
40A9   F1                 POP  AF
40AA   E1                 POP  HL
40AB   D1                 POP  DE
40AC   C1                 POP  BC
40AD   C9                 RET
```

.ISTAGEM HEXADECIMAL

```
16.514     C5 D5 E5 F5 3A 7B 40 2A
16.522     0C 40 23 36 00 11 1F 00
16.530     19 54 5D 01 20 00 09 0B
16.538     ED B8 23 36 00 3D 20 ED
16.546     06 20 36 00 23 10 FB F1
16.554     E1 D1 C1 C9
```

* * * S Q R T * * *

DESCRICAO

Esta rotina calcula a raiz quadrada de um numero de 2 bytes.

CONDICOES DE ENTRADA

O numero de 2 bytes devera estar em 407Bh (16.507) e em 407Ch (16.508), no formato padrao Z-80, primeiro o LSB seguido do MSB.

CONDICOES DE SAIDA

O inteiro da raiz quadrada do numero especificado estara em 407Bh (16.507). 407Ch (16.508) contera 0.

LISTAGEM ASSEMBLY

```
4082  C5          SQRT    PUSH BC
4083  D5                  PUSH DE
4084  E5                  PUSH HL
4085  2A 7B 40            LD   HL,(407B)
4088  06 FF               LD   B,FF
408A  11 FF FF            LD   DE,FFFF
408D  04          SQRT1   INC  B
408E  19                  ADD  HL,DE
408F  1B                  DEC  DE
4090  1B                  DEC  DE
4091  38 FA               JR   C SQRTT1
4093  68                  LD   L,B
4094  26 00               LD   H,00
4096  22 7B 40            LD   (407B),HL
4099  E1                  POP  HL
409A  D1                  POP  DE
409B  C1                  POP  BC
409C  C9                  RET
```

LISTAGEM HEXADECIMAL

```
16.514     C5 D5 E5 2A 7B 40 06 FF
16.522     11 FF FF 04 19 1B 1B 38
16.530     FA 68 26 00 22 7B 40 E1
16.538     D1 C1 C9
```

DESCRICAO

Esta rotina busca um caractere dentro de uma string. Por string entende-se nao somente o conceito do basic de string, podendo ser qualquer area da memoria. Como devemos passar para a rotina o endereco da string, seu comprimento, e o caractere desejado utilizamos para esta finalidade um bloco de parametros.

CONDICOES DE ENTRADA

Devemos colocar em 407Bh (16.507) e em 407Ch (16.508) o endereco do bloco de parametros, no formato padrao Z-80, primeiro o LSB seguido pelo MSB. O bloco de parametros devera ser formado pelo seguinte:

Byte 0:LSB do endereco da string
Byte 1:MSB do endereco da string
Byte 2:LSB do numero de bytes da string
Byte 3:MSB do numero de bytes da string
Byte 4:Caractere procurado
Byte 5:Reservado para o resultado
Byte 6:Reservado para o resultado

CONDICOES DE SAIDA

Se o caractere desejado foi encontrado dentro da string, os bytes 5 e 6 do bloco de parametros conterao o endereco do caractere, no formato padrao Z-80, primeiro o LSB seguido do MSB. Caso o caractere nao seja encontrado os bytes 5 e 6 do bloco de parametros conterao FFFFh. Os primeiros 5 bytes do bloco de parametros, 407Bh (16.507) e 407Ch (16.508) nao sao alterados.

LISTAGEM ASSEMBLY

```
4082  C5             SSOC   PUSH BC
4083  D5                    PUSH DE
4084  E5                    PUSH HL
4085  F5                    PUSH AF
4086  2A 7B 40              LD   HL,(407B)
4089  5E                    LD   E,(HL)
408A  23                    INC  HL
408B  56                    LD   D,(HL)
408C  23                    INC  HL
408D  4E                    LD   C,(HL)
408E  23                    INC  HL
408F  46                    LD   B,(HL)
```

```
4090  23                 INC  HL
4091  7E                 LD   A,(HL)
4092  EB                 EX   DE,HL
4093  ED B1              CPIR
4095  20 03              JR   NZ SSOC1
4097  2B                 DEC  HL
4098  18 03              JR   SSOC2
409A  21 FF FF    SSOC1  LD   HL,FFFF
409D  EB          SSOC2  EX   DE,HL
409E  23                 INC  HL
409F  73                 LD   (HL),E
40A0  23                 INC  HL
40A1  72                 LD   (HL),D
40A2  F1                 POP  AF
40A3  E1                 POP  HL
40A4  D1                 POP  DE
40A5  C1                 POP  BC
40A6  C9                 RET
```

LISTAGEM HEXADECIMAL

```
16.514    C5 D5 E5 F5 2A 7B 40 5E
16.522    23 56 23 4E 23 46 23 7E
16.530    EB ED B1 20 03 2B 18 03
16.538    21 FF FF EB 23 73 23 72
16.546    F1 E1 D1 C1 C9
```

)ESCRICAO

Esta rotina busca uma substring de dois :aracteres dentro de uma string. Por string entende-se nao somente o conceito do basic de string, podendo ser qualquer area da memoria. Como devemos passar para a rotina o endereco da string, seu comprimento, e os caracteres desejados, utilizamos para esta finalidade um bloco de parametros.

:ONDICOES DE ENTRADA

Devemos colocar em 407Bh (16.507) e em 407Ch (16.508) o endereco do bloco de parametros, no formato padrao Z-80, primeiro o LSB seguido pelo MSB. O bloco de parametros devera ser formado pelo seguinte:

Byte 0:LSB do endereco da string
Byte 1:MSB do endereco da string
Byte 2:LSB do numero de bytes da string
Byte 3:MSB do numero de bytes da string
Byte 4:Primeiro caractere procurado
Byte 5:Segundo caractere procurado
Byte 6:Reservado para o resultado
Byte 7:Reservado para o resultado

:ONDICOES DE SAIDA

Se os caracteres desejados forem encontrados dentro da string, os bytes 6 e 7 do bloco de parametros conterao o endereco do primeiro caractere, no formato padrao Z-80, primeiro o LSB seguido do MSB. Caso os caracteres nao sejam encontrados os bytes 6 e 7 do bloco de parametros conterao FFFFh. Os primeiros 6 bytes do bloco de parametros, 407Bh (16.507) e 407Ch (16.508) nao sao alterados.

LISTAGEM ASSEMBLY

```
4082  C5          SSTC  PUSH BC
4083  D5                PUSH DE
4084  E5                PUSH HL
4085  F5                PUSH AF
4086  2A 7B 40          LD   HL,(407B)
4089  5E                LD   E,(HL)
408A  23                INC  HL
408B  56                LD   D,(HL)
408C  D5                PUSH DE
```

```
408D  23              INC   HL
408E  4E              LD    C,(HL)
408F  23              INC   HL
4090  46              LD    B,(HL)
4091  23              INC   HL
4092  5E              LD    E,(HL)
4093  23              INC   HL
4094  56              LD    D,(HL)
4095  E3              EX    (SP),HL
4096  7B        SSTC1 LD    A,E
4097  ED B1           CPIR
4099  20 0B           JR    NZ SSTC2
409B  78              LD    A,B
409C  B1              OR    C
409D  28 07           JR    Z SSTC2
409F  7A              LD    A,D
40A0  BE              CP    (HL)
40A1  20 F3           JR    NZ SSTC1
40A3  2B              DEC   HL
40A4  18 03           JR    SSTC3
40A6  21 FF FF  SSTC2 LD    HL,FFFF
40A9  EB        SSTC3 EX    DE,HL
40AA  E1              POP   HL
40AB  23              INC   HL
40AC  73              LD    (HL),E
40AD  23              INC   HL
40AE  72              LD    (HL),D
40AF  F1              POP   AF
40B0  E1              POP   HL
40B1  D1              POP   DE
40B2  C1              POP   BC
40B3  C9              RET
```

LISTAGEM HEXADECIMAL

```
16.514     C5 D5 E5 F5 2A 7B 40 5E
16.522     23 56 D5 23 4E 23 46 23
16.530     5E 23 56 E3 7B ED B1 20
16.538     0B 78 B1 28 07 7A BE 20
16.546     F3 2B 18 03 21 FF FF EB
16.554     E1 23 73 23 72 F1 E1 D1
16.562     C1 C9
```

A

MICRON ELETRONICA COM. IND. LTDA

CAIXA POSTAL 100

12.200 SAO JOSE DOS CAMPOS SP

SIM! Desejo receber as atualizacoes desta edicao, sem
pesas.

ETENTE..

ERECO..

.... CIDADE...................... ESTADO.......

IPAMENTO UTILIZADO....................................

NTAS PESSOAS UTILIZAM O MESMO EQUIPAMENTO.............

AL DE AQUISICAO DO LIVRO..............................

NDIMENTO RECEBIDO.....................................

../../..   ....

Neste livro você encontrará a rotina que esta procurando para aumentar o desempenho de seus programas, sejam eles escritos em basic ou em linguagem de máquina conforme a relação abaixo:

CBSB : Converte um número binário em string binária
CBSD : Converte um número em string decimal
CBSH : Converte um número binário em string hexadecimal
CHKS : Faz o checksum de uma área de memória
CSBB : Converte string binária em número binário
CSDB : Converte string decimal em número binário
CSHB : Converte string hexadecimal em número binário.
CLSN : Limpa linhas do vídeo
CSTR : Compara strings
DDPO : Divide número de 16 bits por número de 8 bits
DDPD : Divide número de 16 bits por número de 16 bits
DLBL : Deleta bloco da memória
EXOR : EXCLUSIVE OR entre dois números de 8 bits
INST : Entra com uma string a partir do teclado
MDPD : Multiplica dois números de 16 bits
MOBL : Mover bloco de memória
MOPL : Multiplica dois números de 8 bits
MOPO : Multiplica dois números de 8 bits rapidamente
MPAD : Adição em multiplica precisão
MPSB : Subtração em multiplica precisão
MSLF : Múltiplos deslocamento à esquerda em 16 bits
MSRG : Múltiplos deslocamentos à direita em 16 bits
PRME : Preencher bloco de memória
PSLN : Preencher linhas do vídeo com caractere desejado
PTST : Imprimir string no vídeo
RVPB : Rodar o vídeo para baixo
RVPC : Rodar o vídeo para cima
RVPD : Rodar o vídeo para a direita
RVPE : Rodar o vídeo para a esquerda
SCDL : Scroll do vídeo para baixo e para esquerda
SCDR : Scroll do vídeo para baixo e para direita
SCDW : Scroll do vídeo para baixo
SCLT : Scroll do vídeo para a esquerda
SCRG : Scroll do vídeo para a direita
SCUL : Scroll do vídeo para cima e para a esquerda
SCUP : Scroll do vídeo para cima
SCUR : Scroll do vídeo para cima e para a direita
SQRT : Raiz quadrada de um número de 16 bits
SSOC : Busca um caractere em uma string
SSTC : Busca dois caracteres em uma string

SOBRE O AUTOR

Gino Douglas de Carvalho é Engenheiro Eletricista formado pela Universidade de Taubaté, e trabalha como encarregado de Projetos na Johnson & Johnson desde 1977. É professor de linguagem de máquina na Ensicom — Taubaté e atua como acessor em Hardware e Software.